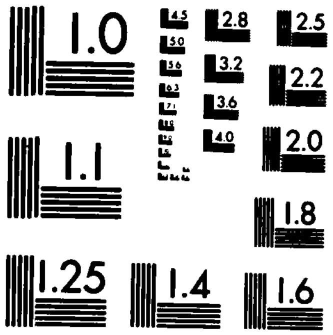
AD-A147 018 A SIMULATION MODEL FOR AIR LAUNCHED CRUISE MISSILE 1/2
ENGINE MANAGEMENT(U) AIR FORCE INST OF TECH
WRIGHT-PATTERSON AFB OH SCHOOL OF SYST..
UNCLASSIFIED D P RICKARD ET AL. SEP 84 F/G 5/1 NL



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

2

A SIMULATION MODEL FOR AIR

LAUNCHED CRUISE MISSILE ENGINE MANAGEMEN

THESIS

Douglas P. Rickard
Captain, USAF

Thomas G. Schommer
Captain, USAF

AFIT/GLM/LSM/84S-55

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

2

AFIT/GLM/LSM/84

A SIMULATION MODEL FOR AIR LAUNCHED CRUISE MISSILE ENGINE MANAGEMENT

THESIS

Douglas P. Rickard Captain, USAF

Thomas G. Schommer Captain, USAF

AFIT/GLM/LSM/84S-55

AFIT/GLM/LSM/84S-55

A SIMULATION MODEL FOR AIR LAUNCHED CRUISE MISSILE ENGINE MANAGEMENT

THESIS

Presented to the Faculty of the School of Systems and Logistics

of the Air Force Institute of Technology

Air University

In Partial Fulfillment of the

Requirements for the Degree of

Master of Science in Logistics Management

Douglas P. Rickard, B.S.
Captain, USAF

Thomas G. Schommer, B.A.
Captain, USAF

September 1984

ACKNOWLEDGEMENTS

We wish to thank our wives Sue Rickard and Elvira Schommer for their infinite patience during the past year. Without their understanding, support and encouragement, this thesis could not have been completed.

We would also like to thank Lt Colonel Peter Bobko for his excellent guidance in the subjects of modeling, simulation and the SIMSCRIPT language, and for guiding the formulation stages of this thesis effort. We also wish to thank Mr. Robert Uhrig for his insights into the Air Launched Cruise Missile system. Finally we are deeply indebted to Dr. Panna B. Nagarsenker for her expert help in completing our analysis and finishing the writing on time.

Douglas P. Rickard
Thomas G. Schommer

Table of Contents

List of Figures

List of Tables

AFIT/GLM/LSM/84S-55

Abstract

This paper examines the management of spares acquisition and logistics support activities for the Air Launched Cruise Missile (ALCM) engine. A SIMSCRIPT II.5 simulation model of the ALCM system is developed and the probable ranges of five relevant factors (transportation time, maintenance duration, engine failure rate, test duration and test loss rate) are determined. The model is manipulated using a factorial design with the five factors at the extremes of their ranges. An ANOVA is used to determine if changes in these five factors significantly impact the operational capability of the ALCM engine. This capability is expressed as the average number of days an engine must be used as an operational asset past the manufacturer's warranty period without an overhaul. The ANOVA indicated that changes in transportation time and maintenance duration had a significant effect on the operational capability of the ALCM engine and thus required further investigation to refine their range of values.

The model's operation, verification, input requirements and output capabilities were documented so that the model could be used as a management tool when validation is completed. This documentation will give other modelers the

depth of understanding necessary to adapt the model to their particular use. The model was designed with the flexibility necessary to modify the assumptions and limitations built into it so the model could "grow" as the ALCM system evolves.

ALCM engine managers should eventually be able to use the model to test the effects (on a number of variables of interest) of changes in maintenance policies and timing. The model should easily be adaptable to reflect the environment of the Ground Launched Cruise Missile and the Submarine Launched Cruise Missile systems. It could then be used to estimate the number of spare engines required in these systems to meet various engine capability levels and other management goals.

A SIMULATION MODEL FOR AIR LAUNCHED CRUISE MISSILE ENGINE MANAGEMENT

I. Introduction and Background

General Issue

The management of spares acquisition and logistics support efforts is a very important part of the procurement and future viability of any weapon system. Acquisition and support decisions significantly impact both system costs and weapon system capabilities. Acquisition costs have increased to the extent that repairable spares contracts frequently involve many millions of dollars. These high costs make it imperative to determine a total system support concept that provides the required system capabilities while controlling spares acquisition costs. However, modern weapon systems are becoming more and more complicated, making the development of a viable support concept more difficult. A greater understanding of the system's operational environment and maintenance requirements is needed as system complexity increases.

The Air Launched Cruise Missile (ALCM) is one of a family of three similar missiles that are currently (or soon will be) employed by all three branches of the armed services. Like most weapon systems, many of the ALCM's parts can be repaired if broken and require routine servicing at specified time intervals. During the time these parts are being repaired, spare parts must be available to replace them to keep the number of deployed missiles at a specified level. For major repairable spare parts, such as the engine assemblies that this paper considers, it is important to find out how many spares will be required over the life of the system before the original production line is closed down. This is the life-of-type procurement strategy for special item procurements as authorized by Air Force Logistics Command (AFLC) regulations (1:1-5). If too few engines are procured, the number of operational missiles will (at some point) fall below the specified level or the production line will have to be opened up again (at considerable expense) to produce more spares. If too many engines are procured, large amounts of money will have to be spent unnecessarily. Effective and economic system support, including the storage, distribution and maintenance of these spare engines is just as critical as the correct number of spares in providing the required system capabilities. Thus the effects of engine failure rates, maintenance duration and frequency, transportation time, and testing requirements on system capabilities must be considered as well as the number of spares.

The responsibilities for managing acquisition programs are directed by Air Force Regulation 800-2, Acquisition Program Management (5). The weapon system acquisition process is managed through a Systems Program Office (SPO) and guided by a Program Management Plan (PMP). A SPO has latitude in developing an effective plan for the particular system.

> A program manager involved in production planning is frequently faced with a high degree of uncertainty surrounding both the timing and the quantity of the requirements for his particular system or subsystem. . . . It is advantageous for the program manager to consider a wide range of feasible alternatives in order to structure a production plan adaptable to changing conditions (4:50).

Current methods used to determine the quantity of spares to purchase for a weapon system during acquisition are governed by Department of Defense Instruction (DODI) 4140.42 (6). This instruction encourages the use of operational demand data as the primary source of input to determine the level of spares required. DODI 4140.42 "allows for sparing of essential items that do not meet the demand criteria (and). . . sparing by alternative computational techniques which minimize system downtime (14:14)." But providing for alternatives does not create valid techniques. "The major criticism in the inventory/supply area for military application is the lack of attention to objectives that emphasize weapon systems availability and capability (7:13)." But a comprehensive spares management methodology for the ALCM engine program does not exist. One of the major difficulties in devising a methodology that considers weapon system availability, along with engine requirements, is the lack of demand data. Since this is the initial deployment of the

ALCM, a data base has not been established. The models developed for DOD use a demand data base for requirements determination, and are not suited for this problem.

Thus one sees that the current analytical methods used to forecast demand for repairable spares and to determine the impact of different support concepts do not apply directly to the ALCM engine system. The complexity and multiplicity of factors bearing on the problem make it difficult enough to even conceptualize the system, much less to apply mathematically tractable analytical techniques to it. As Emory says,

> . . . there are many processes for which there is no analytical solution, or at least not one attainable at a reasonable cost. Often the processes are so complex as to defy analytical solution with the present state of the art. It is in these cases that simulation has the advantage (8:355).

For these reasons a simulation model will be used to examine the logistics requirements for the ALCM engine.

A simulation model can be used to assist the manager in examining and understanding the system under study and in determining the significant relationships which should be included in planning for overall system support. A model provides a cost effective means of considering numerous feasible alternatives to complex acquisition and logistics management situations (17:11). Models record the important elements of a system, providing the manager with a tool with which to objectively view the parts and their interactions.

A model aids the basic decision process and helps a manager determine the spares level and support requirements to use.

Drezner and Hillestad in their report titled "Logistics Models: Evolution and Future Trends" state:

> There should be more effective use of good support models in the ongoing management of logistics. We must move from the bean counting approach of current readiness reporting to using models to predict force capability to go to war (7:20).

Even if there is no past demand data with which to work, a simulation model can be a good tool in understanding future system requirements, if the model is a good representation of the actual system. A limit to the capabilities of simulation is expressed by Emory :

> We can not identify the optimal answer to this problem with a simulation. However, if we rerun the simulation a number of times with different . . . strategies, we can identify the best of the strategies that we test (8:361).

The value of this tool depends on the validity of the assumptions made about the system and how accurately the model has captured the important relationships between the determining variables in the system.

Specific Problem

The Air Launched Cruise Missile (ALCM) is now being deployed as an operational missile at various bases. ALCM system managers need information of the effects on ALCM operational capabilities of 1) the number of spare engines procured 2) maintenance and testing policies 3) engine failure rates and 4) transportation times. A need exists for a model that will estimate the effects on ALCM operational capabilities of the number of spare engines procured and the logistics policies used to manage them. The model must be documented as to its assumptions and the significance of its output. Information as to which parameters could be adjusted to reflect more current information must be incorporated in this documentation. As Shannon states:

> No simulation project can be considered successfully completed until it has been accepted, understood, and used. . . . Careful and complete documentation of the development and operation of the model can greatly increase its useful life and chances of successful implementation (17:32-33).

The model must reflect the random nature of many of the factors from which it will be derived, and validity must be established for its proposed application. Documentation and validation will provide information on how management should treat the model's output (how management might best use the information the model develops) and the confidence management should have in this output.

System Overview

Figure 1 is a diagram of the ALCM system showing the major factors that impact system support requirements for spare engines. Spare engines are required to fill the transportation pipeline of repaired engines between the depot and operating base, broken engines being shipped to the depot, and those engines being shipped for periodic

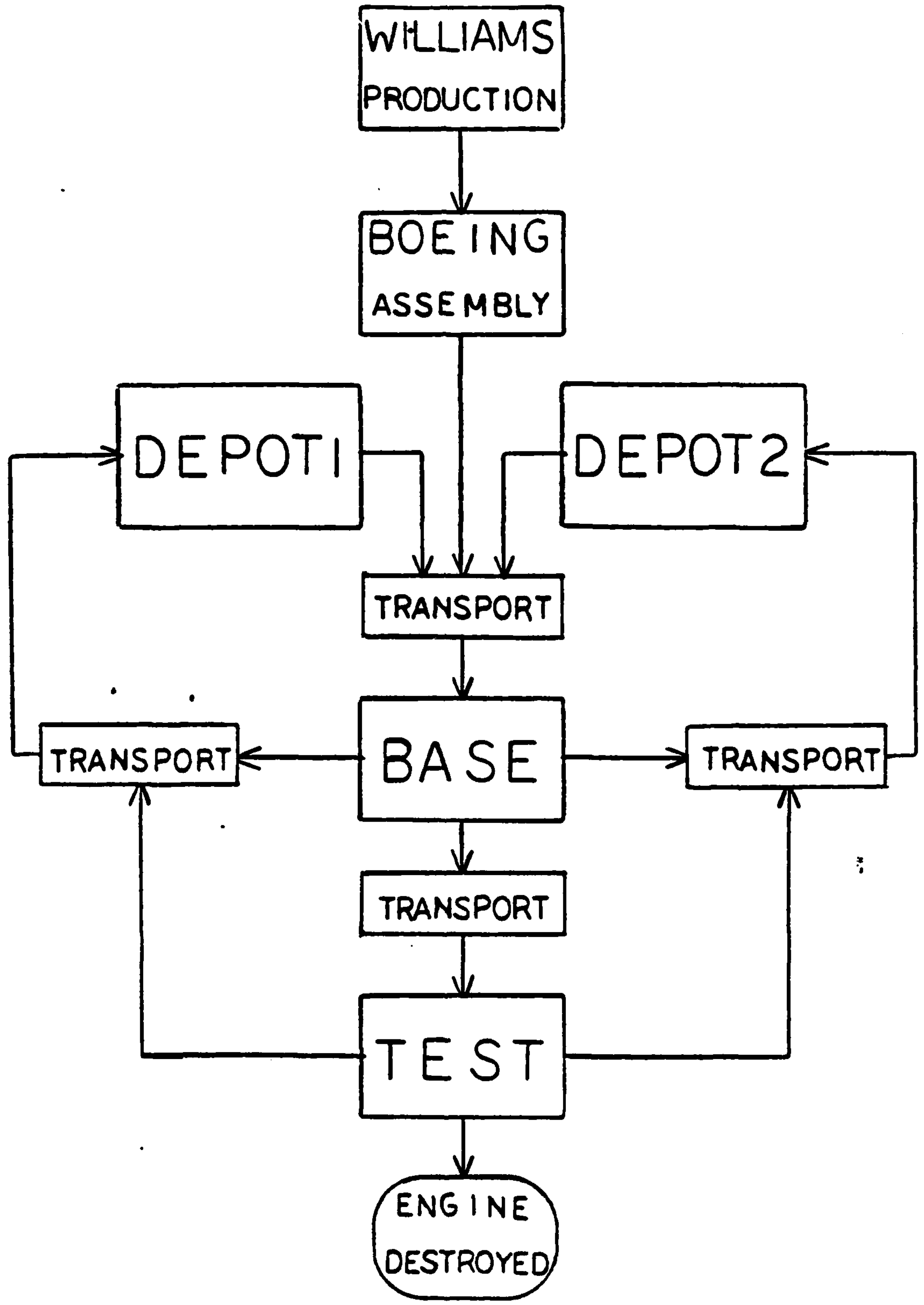


Figure 1. System Diagram

depot maintenance (21:14,20). Spare engines are also required during the time the engines are undergoing depot maintenance or planned modifications. Finally, every new weapon system undergoes acceptance and improvement tests for many years after initial deployment (20). These tests are usually run on systems or components which are part of operational stocks. For the ALCM, engines at operational bases are selected for testing, transported to the test location, and then undergo one of several performance tests. One of these tests will require a spare engine to replace the test engine taken from the operating base.

The Williams International plant manufactures ALCM engines and provides these engines to the Boeing facility where they are mated with airframes, the production process is completed, and the missile shipped to the operating base. In addition to these engines, Williams will provide a specified number of spare engines to be used in the ALCM system. Williams will also perform primary depot level repair and scheduled maintenance on the engines (which is time-phased). Each engine is warranted for a certain time period after its original date of manufacture and after each subsequent overhaul. The extent of depot maintenance is determined by the type of engine (F107-101 or F107-104) and the phase of maintenance due (limited or full). After the first warranty period has elapsed, the engine will be due a limited overhaul. After the second warranty period, it will be due a full overhaul, then the cycle repeats itself.

Refurbishment is another type of overhaul done at Williams to return an engine used in testing to a serviceable condition. A final servicing category is the conversion from the F107-101 type engine currently being produced to a F107-104 type to provide more power and extend the engine's warranted service life. This type-conversion will take place from 1988 to 1992 at the Williams International depot facility. The alternate engine repair and servicing facility (depot), at Oklahoma City Air Logistics Center (OCALC), is scheduled to begin operations in 1989. This facility will perform limited overhauls only.

The deployment bases are where the missiles are scheduled to be in operational use. Missiles will initially be transported from the assembly plant at Boeing to the operational bases. Once a missile arrives on a base, it is used as an operational asset: in storage, on alert, etc . The engine's warranted life (30 months for a F107-101 or 60 months for a F107-104) begins when it is delivered to the Boeing plant for missile assembly and ends 30 or 60 months after this date. This warranted life is renewed upon completion of depot servicing and, again, ends 30 or 60 months after this date. The engine will be removed from the missile for overhaul when its warranted life has expired and a spare engine is available to replace it. If an earlier requirement for an engine exists to smooth the depot work flow, an engine may be removed before its warranted lifetime has expired. Although warranted for the length of time cited above, the engine may fail prematurely and require replacement. All engines in which a failure is detected will be removed from alert duty and replaced as soon as a spare engine is available. Engines selected for testing will also be removed when a spare is available. Complete missiles selected for testing will not require a spare engine, and will be returned to their respective bases after refurbishment. Engines removed for failure or warrantee expiration are sent back to the depot for overhaul, while those removed for testing, along with selected test missiles, are sent to the appropriate test facility.

The test types include: Engine Verification and Improvement Program (EVIP) - a non-destructive test on a removed engine, Operational Test Launches (OTL) - a free flight test of a complete missile with a planned midair recapture (there is a possibility of the missile not being recaptured and thus being destroyed in this test), and Joint Test Assembly (JTA) - a free flight test of a complete missile resulting in missile destruction. The OTL test keeps a complete missile unavailable for the duration of the test. The EVIP test keeps only the missile's engine unavailable for the duration of the test. After testing, each engine or surviving missile is sent to the depot for refurbishment, and then redistributed as a spare engine or a replacement missile. The exclusion of the Production Assurance Test (performed on engines prior to their assembly), will not affect the model's validity. An excess of engines over airframes exists and will continue to exist, so the test will not delay missile availability.

Research Objectives

The objectives of this research include : 1) A thorough review of the ALCM spare engine system to determine the system's relevant variables and their interrelationships, and the amount and type of variability in these relationships. 2) Building a computer simulation model of the system will then be developed to reflect these findings. 3) Experimenting with the model to assess the significance of changes in these variables on the system's operational capability. 4) Documenting the derivation, use, applicability and significance of the model's output. . 5) Incorporating flexibility into the model's design to allow for changes that will invariably occur with increased knowledge of the real system. This flexibility must also allow the assessment of changes to system variables not addressed in this research, and make the model adaptable to changes in management perspective.

Research Question

A final objective of this research is to answer three specific research questions: Do any of the selected variables (see factor variability pages 19-22) have a significant effect on the ALCM engine operational capability when allowed to vary within their probable range of values? Which of the variables are significant? What are the implications of these findings? ALCM engine operational capability is the overall probability of a given missile's engine successfully firing and operating as designed when a demand is placed on it. This capability assumes the system is able to maintain a specified (classified) number of operational missiles at all times. For this research, engine operational capability will be expressed in terms of the average time (in days) that an engine's operational life (usage after manufacturing or depot servicing) exceeds its warranted service life. This operational definition assumes a decrease in the probability of firing when an engine exceeds its warranted life. The particular relationship between engine life and probability of firing on demand is not known at this time. Due to the limited time and resources available for this research, this relationship will not be addressed in this paper, though this relationship would provide a great deal of useful information. This paper will assume that management can use the stated measure as a viable proxy for the overall probability of an engine firing and operating successfully.

Limitations and Scope

Some factors exist in the ALCM system that will not be examined in this model, therefore the model will be developed with the following assumptions. Exceeding the engine's warranted life increases the probability of an engine not firing on demand. The engine's warranted service life and its probability of premature failure is independent of the type service it sees (storage, alert duty, captive flight, etc.), which is a reflection of the missile's use. The engine will never be used, run up or tested at base level unless it is actually engaged against a target, and at that time, is unrecoverable (20).

Once an engine is determined to be destined for destructive testing, it is considered out of the system at that time. When a missile is selected for testing, all the various test procedures will be considered as one factor. Thus the model will not reflect the individual variables at a test site, but will consolidate them into one variable, the average time required to accomplish the test.

The same is true of the depot level maintenance function, the relevant factor in this model being the time an engine spends in the depot, in total, not the actual process it goes through while at the depot. Furthermore, all spare engines, except those destructively tested, can be renewed indefinitely (with appropriate servicing to "as good as new" condition). That is, once an engine is produced, it is always a potential spare resource. In addition , each depot is considered to have an unlimited servicing capacity (each depot can service an unlimited number of engines at the same time) (20).

This model will be designed, however, with the flexibility necessary to modify assumptions and limitations, and insert newly discovered relevant variables or different levels of factors, as management may see fit, at any time in the future. Since "every model is based upon certain assumptions regarding an uncertain future which the model is supposed to organize and eventually predict . . . (16:293)", the model must be able to be changed when the assumptions change or are proved invalid.

The main thrust of this model will be to give management a tool to both understand the ALCM system and to determine the significance that changes in selected variables have on system capabilities. Those variables found to have a significant impact on system capabilities can be investigated in depth to determine their actual range of variability. Control efforts can then be directed toward these factors, reducing the emphasis placed on those which do not affect system capabilities. This will allow managers to expend their resources where they will be the most productive. In addition, the model will be constructed to allow forecasts of the monthly demand for spare engines, and monthly requirements for the five types of depot servicing. It will also be able to estimate the effects of various maintenance, test, and transportation scenarios on system capability. The model will have the flexibility to improve with age (and better data) for increasingly accurate forecasts.

II. Methodology

The methodology that will be used in this research is an expansion of the Systems Science Paradigm as expressed by Schoderbek, et al, which is an "application of the systems approach to the study of real-world phenomena (16:295-304)". The System Science Paradigm consists of three successive phases: Conceptualization, Analysis and Measurement, and Computerization.

Conceptualize the Problem

The first step will be the conceptualization of the problem, defined by Schoderbek, et al. as

> understanding and organizing the interactions among the elements making up the phenomenon under scrutiny into a logical network of relationships in such a way as to reveal the direction of the underlying structure (16:290).

The purpose of the system being modeled must be thoroughly understood. Then the modeler can clearly state the objective of his or her study. The variables which interact within the system, and between the system and its environment, must be examined. The model should include only those independent variables determined to be relevant to the accomplishment of the stated objectives. The model should be structured to permit the measurement of the dependent variables to determine whether or not the stated objectives have been met (16).

TABLE I

ENGINE PRODUCTION SCHEDULE

YEAR	JAN	FEB	MAR	APR	MAY	JUN	JUL	AUG	SEP	OCT	NOV	DEC
1981	0	0	0	0	0	6	8	8	12	12	14	17
1982	22	24	28	32	35	40	40	40	40	40	40	40
1983	40	40	40	40	40	41	39	40	40	40	40	40
1984	40	47	47	26	35	27	28	27	28	27	28	27
1985	27	27	25	30	21	16	17	19	20	20	20	20
1986	20	20	21	26	1	0	0	0	0	0	0	0

TABLE II

MISSILE PRODUCTION

YEAR	JAN	FEB	MAR	APR	MAY	JUN	JUL	AUG	SEP	OCT	NOV	DEC
1981	0	0	0	0	0	0	0	0	0	0	2	5
1982	7	9	12	15	18	22	28	32	35	40	40	40
1983	40	40	40	40	40	40	40	40	40	40	37	37
1984	37	36	37	36	36	36	36	36	36	40	28	28
1985	28	28	28	28	27	27	27	27	27	27	20	20
1986	20	20	20	20	20	20	20	20	20	20	0	0

Data Collection. The background information used to describe the ALCM system as well as a detailed description of variables in the system were provided by the Engine Logistics Office of the Aeronautical Systems Division (ASD) at Wright-Patterson AFB, Ohio (20). ASD provided a regular engine production schedule for the Williams plant, which began in June 1981 and extends through April 1986. This schedule is depicted in Table I, which shows the number of engines produced per month over this five year period. A total of 1715 F107-101 engines will be produced during this period. Table II shows the missile production schedule. This schedule depicts the number of complete missiles (engine and airframe) assembled and deployed per month from November 1981 to October 1986.

A decision was made in January of 1984 to procure 115 spare F107-101 engines for the ALCM system. The spare engines were produced at the Williams production facility from January 1982 to December 1983 concurrently with the regular engine production run.

TABLE III

EVIP TEST SCHEDULE

Days past 1 January 1981

570	1311	1887	2463	3039	3687	4263	4839	5415
770	1347	1923	2499	3075	3723	4299	4875	5475
810	1383	1959	2535	3111	3759	4335	4911	5520
850	1419	1995	2571	3147	3795	4371	4947	5565
890	1455	2031	2607	3183	3831	4407	4983	5610
930	1491	2067	2643	3219	3867	4443	5019	5655
970	1527	2103	2679	3255	3903	4479	5055	5700
1010	1563	2139	2715	3291	3939	4515	5091	5745
1050	1599	2175	2751	3363	3975	4551	5127	5790
1090	1635	2211	2787	3399	4011	4587	5163	5835
1095	1671	2247	2823	3471	4047	4623	5199	
1131	1707	2283	2859	3507	4083	4659	5235	
1167	1743	2319	2895	3543	4119	4695	5271	
1203	1779	2355	2931	3579	4155	4731	5307	
1239	1815	2391	2967	3615	4191	4767	5343	
1275	1851	2427	3003	3651	4227	4803	5379	

Tables III through V show representative schedules for the various tests to be performed on engines and complete missiles beginning in 1982 and continuing into 1996 (in days past 1 January 1981). These tests will be conducted in the year they are scheduled, though the specific dates will vary.

TABLE IV

OTL TEST SCHEDULE

Days past 1 January 1981

545	1365	1860	2310	2795	3235	3675	4150	4745
730	1410	1905	2355	2835	3275	3715	4195	4866
803	1455	1950	2400	2875	3315	3755	4240	4987
876	1500	1995	2445	2915	3355	3795	4285	5108
949	1545	2040	2490	2955	3395	3835	4330	5229
1022	1590	2085	2555	2995	3435	3875	4380	5350
1095	1635	2130	2595	3035	3475	3915	4440	5471
1140	1680	2175	2635	3075	3515	3955	4500	5592
1185	1725	2200	2675	3115	3555	4015	4560	5713
1275	1770	2220	2715	3155	3595	4060	4620	
1320	1815	2265	2775	3195	3635	4105	4680	

TABLE V

JTA TEST SCHEDULE

Days past 1 January 1981

730	1168	1642	2097	2676	3402	4128	4854	5580
803	1241	1690	2188	2797	3523	4249	4975	
876	1314	1733	2279	2918	3644	4370	5096	
949	1387	1824	2370	3039	3765	4491	5217	
1022	1460	1915	2461	3160	3886	4612	5338	
1095	1551	2006	2555	3281	4007	4733	5459	

Table VI shows a proposed schedule (developed by the authors) of engine type conversions (F107-101 to F107-104). This is the modification of the original engine to the upgraded version. This schedule is based on the attempt to modify the maximum number of engines possible during their regularly scheduled "full" overhaul, given the constraints of 380 modification kits being available in 1988, 420 in 1989, 510 in 1990, and 568 in 1991. This schedule is based on projected overhaul dates stemming from the engines' initial production dates, but does not account for engine failures earlier than the warranted time, nor for cycle changes caused by engine testing. It attempts to minimize conversions of limited overhauls in 1988-1989 that could be converted during their scheduled full overhauls in 1990-1991. Since the conversion process takes the same amount of time as a full overhaul, but two-to-five times as long as a limited overhaul, this schedule would tend to decrease the total time engines would spend in the depot over the conversion period.

TABLE VI

CONVERSION SCHEDULE

NUMBER OF FULL CONVERSIONS ALLOWED

YEAR	JAN	FEB	MAR	APR	MAY	JUN	JUL	AUG	SEP	OCT	NOV	DEC
1988	21	45	45	45	45	46	0	0	0	0	0	0
1989	0	0	0	19	35	27	28	27	28	27	0	0
1990	28	27	0	0	21	16	17	19	20	20	20	20
1991	20	20	21	26	0	6	8	8	12	12	14	17

NUMBER OF LIMITED CONVERSIONS ALLOWED

YEAR	JAN	FEB	MAR	APR	MAY	JUN	JUL	AUG	SEP	OCT	NOV	DEC
1988	0	0	0	0	0	0	28	27	25	30	23	0
1989	0	0	0	0	0	0	27	29	32	37	40	44
1990	45	45	45	45	45	43	24	0	0	0	0	0
1991	44	45	45	45	22	43	40	47	47	7	0	0

<u>Factor Variability</u>. From the information gathered from ASD during the initial interview, a flow model was developed by the authors. This model depicted the flow of the engines through the projected ALCM system as the authors initially conceptualized it. Subsequent interviews and questions

about unclear relationships and variable values helped refine and correct this model. The ASD personnel validated the final flow model as an accurate reflection of the system as they knew it. The final model depicted five variables which appeared to have the potential for significant impact on ALCM system capabilities, and displayed some variability in their estimated values.

The first variable was the transportation time required to move an engine between the depots and each base or test facility, and the transportation time required to move an engine between a base and the test facilities. The figure offered as the "standard" transportation time (a maximum time in which an engine should arrive at its destination) by ASD/YZL was eight days. This standard time does not account for difference in transportation distance, mode, or possible expediting actions. The shortest time considered likely was a minimum of four days (20). The difference of four days may seem inconsequential. However, considering the total number of trips to be made , it may prove significant (1830 engines x 2 trips per 45 months during 20 years of system life x 4 days per trip = 87,840 days of transportation time difference).

The second variable is the amount of time required to test an engine or missile. Initial estimates indicated sixty days would be required, but later tests were being completed closer to forty days (20). With a potential 20 day difference and a total of 235 EVIP and OTL tests to be completed, engines may be tied up in tests 4,700 days less if the lower figure is true.

The third variable considered is the loss rate for OTL inflight recoveries. Initial estimates indicated a 25% loss rate for OTL tests, but crew experience and improvement in techniques could move this figure closer to 10% early on (20). Since OTL missiles do not require spare engines there is no impact on the number of spare engines required. However, there is an impact on the competition for depot services and the future demand for spare engines if more of the OTL missiles are recovered than expected.

The fourth variable is the amount of time required for depot servicing. the contract negotiated with Williams allows a thirty day period for limited overhauls and a sixty day period for full overhauls. However, investigation of workcards and interviews with William's supervisory personnel by ASD/YZL indicated the possible completion of limited overhauls in as few as six days and full overhauls in thirty days (20). Tests refurbishment, broken engine repair, and F107-101 to F107-104 conversions require approximately the same range of operations as a full overhauls and thus will take approximately the same amount of time. The difference between these two estimates amounts to approximately 400,000 repair days over a 20 year period.

The fifth and final variable is the premature engine failure rate for deployed missiles. Estimated by ALCM system engineers at 15%, this failure rate has not yet been verified by test results (20). Even if 15% is a valid figure for the actual failure rate, its effect on the demand for spare engines may not approach anywhere near this figure. The ALCM engine is maintained in a sealed container during storage, and even when installed in a missile airframe, a failure would be very difficult to detect. Since no field tests for proper operation are conducted at the base (which would involve running the engine, designed essentially for one-time use), the only indication of a failure is by visual, external inspection. If the engine is leaking oil, hydraulic fluid, or physically falls apart, the failure could be detected. Otherwise, a failed engine would remain in a deployed status and assumed to be operational. The undetected failed engine would not require a replacement, resulting in a decreased demand for spare engines. This would effectively reduce the failures close to zero. The distribution of time before failure is unknown and will be assumed to be uniform over the engine's warranted life. With an average loss of 456 days per failed missile, a reduction of failures from 15% to 0% could save almost 1,000,000 days of warranted life before overhaul over a 20 year period.

Factors that will be fixed at a given level are the warranted life for the F107-101 engine (913 days) and the warranted life for the F107-104 engine (1826 days). The number of operational bases will be fixed at six, with five bases requiring 286 missiles and the sixth requiring 285 missiles.

Analysis and Measurement

In the analysis and measurement step, the parameters (those quantities "to which the operator of the model may assign arbitrary values (17:15)") of the model are determined and a parametric model is built. An experimental design is developed specifying how the factors in the model are manipulated and what levels of the factors are studied. Finally, the criteria used (a manager-researcher perspective) to judge the significance of changes in the model's output is stated.

The experimental design of this thesis allowed the researchers to determine which factors or combination of factors significantly affect the model's response measurement. By isolating the factors at both extremes of their hypothesized range, the changes in the model response measurement attributed to changes in each factor's level were determined. Five factors were considered variable, the others were considered as environmental factors or fixed inputs into the system. For the initial screening of these factors, a factorial design was run, varying these factors at two levels, which indicated how sensitive the model was to each of these factors. A factorial design with five factors and two levels required thirty-two replications of the

TABLE VII

VARIABLE FACTORS		
FACTORS	LEVELS	
	Lo	Hi
A. Engine Failure Rate	0	.15
B. Transportation Time (in days)	4	8
C. Test Duration (in days)	40	60
D. Maintenance Duration (in days)		
Limited	6	30
Full/Refurb	30	60
Conversion	30	60
E. OTL Loss Rate	.1	.25

COMBINATIONS	FACTORS				
	A	B	C	D	E
1	Lo	Lo	Lo	Lo	Lo
2	Lo	Lo	Lo	Lo	Hi
3	Lo	Lo	Lo	Hi	Lo
4	Lo	Lo	Lo	Hi	Hi
5	Lo	Lo	Hi	Lo	Lo
6	Lo	Lo	Hi	Lo	Hi
7	Lo	Lo	Hi	Hi	Lo
8	Lo	Lo	Hi	Hi	Hi
9	Lo	Hi	Lo	Lo	Lo
10	Lo	Hi	Lo	Lo	Hi
11	Lo	Hi	Lo	Hi	Lo
12	Lo	Hi	Lo	Hi	Hi
13	Lo	Hi	Hi	Lo	Lo
14	Lo	Hi	Hi	Lo	Hi
15	Lo	Hi	Hi	Hi	Lo
16	Lo	Hi	Hi	Hi	Hi
17	Hi	Lo	Lo	Lo	Lo
18	Hi	Lo	Lo	Lo	Hi
19	Hi	Lo	Lo	Hi	Lo
20	Hi	Lo	Lo	Hi	Hi
21	Hi	Lo	Hi	Lo	Lo
22	Hi	Lo	Hi	Lo	Hi
23	Hi	Lo	Hi	Hi	Lo
24	Hi	Lo	Hi	Hi	Hi
25	Hi	Hi	Lo	Lo	Lo
26	Hi	Hi	Lo	Lo	Hi
27	Hi	Hi	Lo	Hi	Lo
28	Hi	Hi	Lo	Hi	Hi
29	Hi	Hi	Hi	Lo	Lo
30	Hi	Hi	Hi	Lo	Hi
31	Hi	Hi	Hi	Hi	Lo
32	Hi	Hi	Hi	Hi	Hi

experiment. The five factors and their levels for each of the thirty-two experimental runs are shown in Table VII. An analysis of variance (ANOVA) was run on these data points to determine the significance of the main and interaction effects on the primary system measurement (the average time an engine spends in an operational status past its warranted lifetime). The significance of changes in the dependent variable were examined at the alpha = .10, .05 and .01 levels.

Additional information on system performance that the model can produce will be provided in Appendix A: Simulation Output Summary. No formal analysis will be performed on the data presented, but the output is useful in seeing the type of information that the model can provide to system managers.

Computerizing the Model

The model was written in the SIMSCRIPT II.5 simulation programming language, then compiled and run on the CDC CYBER computer system at Wright-Patterson AFB, Ohio. There are four main reasons why SIMSCRIPT II.5 was chosen as the language to model the system. 1) SIMSCRIPT has an English like syntax which simplifies the explanation of the program to managers and other users without a computer programming background. 2) SIMSCRIPT has more capabilities in simulating systems than any other language (17:140). 3) Expertise in SIMSCRIPT was available. 4) SIMSCRIPT allows the modeling of a system on a modular basis. It allows the designing and testing of each module for correct operation independent of all other modules. After individual testing, all the modules can be combined to reflect the workings of the system as a whole. This capability is very important when designing models of complex systems, as it greatly eases model debugging (finding and correcting errors) and verification.

Model Development. The SIMSCRIPT model was designed to parallel the structure of the ALCM system as closely as possible. This is facilitated by the use of the simulation concepts known as "sets", "entities", "processes" and "routines", and by control of these concepts by the "system" and other "owning" entities (see Appendix B: Glossary of Selected SIMSCRIPT Terms). The model's built in "system" controls a series of "sets" which serve as storage facilities for a given class of "entities" (the spare engines and missiles in the ALCM system). One class of entities may represent engines which have been produced by Williams, are stored at the Boeing facility, and have not yet been mated with an airframe. Another class may be those missiles that are operational assets at a certain base. Yet another class may be those engines that have failed or have exceeded their warranted life on base and are awaiting a spare engine to replace them. Sets are able to sequence engines, say, into the depot for repair, based on a set of priorities that the system has established. These sets are also able to perform a variety of record keeping functions, such as: keeping track of how many engines are in a certain class at any particular time (e.g., how many engines are currently being tested), which particular engines are in a given class (e.g., is engine # 344 in an operational missile or is it removed and awaiting repair ?), or which engine of a given class has the highest priority for an available resource (e.g., which engine on which base has been broken the longest ?).

Besides the "system", SIMSCRIPT allows the model to have other "owning" entities. An example that parallels the ALCM system is the two depots that exist in this model. Each of the depots in the model own a set (stockpile) of repaired engines that are available as spare resources. The depots process demands for spare engines, select the oldest engine available, ship the engine to the requesting base, update their inventory records, and inform all interested agencies in the system of the number of spare engines remaining. The depot also informs the system of the repair capability it has remaining, receives and queues incoming engines for servicing, examines the records accompanying the engine to determine the type of servicing it requires, services the engine as indicated, updates the engine's records, and stores the repaired engine in its stockpile.

The model also "creates" the required number of operational "bases", giving them sets to store operational missiles, missiles destined for testing, and engines needing repair or scheduled servicing. The base monitors the status of each missile, detecting its failure or determining its requirement for scheduled servicing. It then requisitions a spare engine if needed, or arranges for shipping a missile to the test site. The base receives incoming spare engines, removes and replaces the old engine, ships the old engine to the appropriate depot for servicing, and updates the records of the old and new engines.

The ALCM system's engines are represented in the model as "temporary entities". Each entity begins its existence when created by an engine production "process", according to a production schedule read into the model from an external file. Each engine carries a set of "attributes" (its own set of records) that indicate a variety of information, such as: the date it was produced or overhauled, when its next servicing is due, what type of servicing it will require, were it is located, its identification number, its precedence for resources as compared to other engines in the same class, which class(es) it belongs to, and many other pieces of information.

The system, depots and bases perform their various functions and make decisions by the use of different "processes". Processes are actions that are scheduled to occur by the system at various times and under various circumstances or combination of circumstances. For example, if an engine at an operational base fails, the base will detect this, and schedule an "engine requisition" process to occur. The requisition process will examine appropriate variables in the system to determine if the engine has indeed failed, a spare engine is available to replace it, and no other engine has a higher priority for replacement. If the appropriate conditions have been met, the requisition process will request the spare engine from the depot, arrange for transporting the spare to the base (via a "transportation process") and schedule an "exchange process" to handle the engine removal and replacement when the spare engine arrives at the base.

Processes accomplish these actions by relying on programming code that reflects various decision rules that are examined whenever more than one course of action is possible. These decision rules are based on the assumptions made about the ALCM system and the priorities established for resource allocation. These rules are designed to parallel the priorities, policies, and decision logic that exists in the actual ALCM system. For a detailed analysis of the decision logic built into the model, refer to Appendix C: Model Operation and Decision Logic.

Some of the many functions that processes perform include: manufacturing engines and scheduling their delivery; gathering statistics on the variables in the system; deploying operational missiles to bases; scheduling and conducting tests; scheduling and performing preventive maintenance, repairs, and modifications; choosing the depot and engine to be used for filling a requisition; choosing the depot to receive and service each engine; and selecting engines for testing.

<u>Model Construction</u>. Model construction began with the "creation" of the physical facilities that exist in the ALCM system: the bases, depots, test facilities, engine storage facilities, etc. These facilities were then given the capability to receive, classify, process, store, and monitor engines (via the assignment of the appropriate sets to each facility). Next, processes were created to handle all the necessary decision making, prioritization, resource allocation, record keeping, and other managemen. functions.

A tree diagram was then developed showing all the possible paths through the system an engine could take, under all combinations of circumstances. The processes were given decision rules to follow at each node for each branch of the tree an engine could follow. The processes were designed to examine the status of the engine (its attributes), the applicable variables in the system, the resources available, the priorities in effect at that time, and the current simulation time before deciding what path the engine would take at each node.

Processes were designed to apply the appropriate probability for path selection if the path to take was subject to chance. This was accomplished by sampling from a built-in (SIMSCRIPT-supplied) random number generator and applying the results to the appropriate probability distribution.

Sets and engines were given attributes, continually updated by the processes, that reflected the current and cumulative status of various output variables. Routines were designed to sample the values of these attributes and other system variables at appropriate times (continuously, daily, monthly, etc.), accumulate and calculate statictics on these values, format the statistics, and print them to an external file for later examination.

Input Data Manipulation. Schedules and variable values were designed to be read by the model from external files prior to starting the simulation (see Appendix D: Input Files SIMU7 and SIMU9, and Tables I through VI) Thus the same model accepts a variety of input parameter values, runs the simulation on the basis of these values, and returns a statistical summary of the simulation to external files. Different runs are easily accomplished with the same model by making a one or two line change of those input variables that differ from one simulation run to the next. Each run's output is directed to a separate output file. After running all the variable combinations desired, the output variables under study are consolidated into one file that is used as input to a statistical analysis program.

Program Documentation. The SIMSCRIPT program developed in the research effort is listed in Appendix E: Program Listing. A more detailed explanation of the SIMSCRIPT program terms is contained in Appendix F: Explanation of Program Terms. This appendix gives a description of each variable's usage, an overview of each processes' function, and details on the functions of the model's sets, attributes and entities. Appendix C: Model Operation and Decision Logic, examines some of the assumptions made in developing the model.

Model Verification

Model verification is the process of "insuring that the model behaves the way an experimenter intends (17:30)". To verify this model the following procedures were used: The flow of an engine through the model was traced to ensure that the intended decision logic was followed. System variables were recorded both before and after a decision point. The variables recorded before the decision point were examined to determine what decision logic should be followed at that point. Then the variables recorded after the decision point were examined to determine if the appropriate decision logic was actually followed. This procedure was repeated tracing different engines until all possible decision logic had been verified throughout the model. An example of this procedure is presented in Appendix G: Sample Verification Process.

Model Validation

Validation "is the process of bringing to an acceptable level the user's confidence that any inference about a system derived from the simulation is correct (17:29)". The importance of model validation must be recognized. The use of an unvalidated model may result in the manager placing his trust in a model that, though verified for correct operation, may not adequately capture the real world phenomena and relationships it is designed to reflect. The complete validation of this model is beyond the scope of this research. Only preliminary validation was accomplished during this research effort. This preliminary validation consisted of an informal review of the output data by the office of ASD/YZL. This office judged the model's output to be a reasonable reflection of what the real-world system would generate, based upon their knowledge of the system at that time. The authors suggest that complete validation occur before managers give full weight to the model's output. Since there is no current data on which to base this validation, the authors suggest that the expert method of validation be used. This involves breaking down the output at different stages, and letting the current experts on those stages pass judgement on whether or not the model's output reflects the output they would expect the real-world system to generate. If not, the experts should assist the modelers in determining what the real output should look like, and where and what the modelers should change to better capture the real situation. Model verification and validation is well covered in the literature by authors such as Berman (2), Garratt (9), Gilmour (10), and Nolan (13).

III. Analysis and Measurement

Introduction

The previous chapters have thoroughly described the operational environment of the ALCM engine system. The authors researched the system through the Engine Logistics Office of the Aeronautical Systems Division (ASD/YZL). A conceptual model was developed from this research and verified for accuracy by this office. An experimental design was developed that would allow the modeler to test the effects of changes in the selected independent variables on the model's dependent variable: the average number of days an engine's operational life exceeds its warranted life. A computer model, written in the SIMSCRIPT II.5 programming language, was then developed and verified. This model reflected the conceptual model's structure and was capable of monitoring changes in the dependent variable brought about by manipulations of the independent variables.

An Analysis of Variance (ANOVA) was used to determine if changes in the independent variables significantly changed the dependent variable. The ANOVA analysis of the output of a factorial design is useful in that all the main effects and interactions of the independent variables can be estimated at the same time (12:3-19).

One of the benefits derived from experimentation with simulation models is that sampling from the population of a model's output can be manipulated by the experimenter "without introducing bias in the response of interest (11:72)". This is done by using Variance Reduction Techniques (VRTs), which reduce "the variance of the estimator by replacing the original sampling procedure by a new procedure which yields the same expected value but with a smaller variance (11:105)" and increase the "reliability of the estimated response of a particular system (11:200)".

A reduction in variance for this analysis was obtained by the VRT of "Common Random Numbers" (11:200-206). This technique uses the <u>same</u> stream of random numbers for each of the model's stochastic variables from run to run. This procedure results in a series of correlated dependent response variables. In investigating the effects of different levels of input variables on the output response variable, one is "not interested in the <u>absolute</u> values of the system responses but in the <u>difference</u> among system responses (11:200)". Since the variance for the difference of two responses, x and y, is given by the equation Var(x-y) = Var(x) + Var(y) - 2 Cov(x,y), any increase in the covariance term will result in a decrease in the variance for the difference. The correlated responses obtained in this research result in a positive covariance term reducing the variance of the dependent variable (11:200). The correlation of observations caused by common random numbers violates the assumption of independent observations normally required for an ANOVA. However, the t-test is considered

TABLE VIII

SIMULATION OUTPUT

A0															
B0								B1							
C0				C1				C0				C1			
D0		D1		D0		D1		D0		D1		D0		D1	
E0	E1	E0	E1	E0	E1	E0	E1	E0	E1	E0	E1	E0	E1	E0	E1
0.00	0.00	10.34	9.68	0.00	0.00	10.94	10.27	0.03	0.00	36.11	35.26	0.23	0.01	36.62	36.28

A1															
B0								B1							
C0				C1				C0				C1			
D0		D1		D0		D1		D0		D1		D0		D1	
E0	E1	E0	E1	E0	E1	E0	E1	E0	E1	E0	E1	E0	E1	E0	E1
0.00	0.00	19.09	20.88	0.00	0.00	19.74	20.79	0.02	0.01	52.70	52.09	0.03	0.02	53.15	53.14

robust enough to indicate significance even when independence assumptions are violated (3).

Testing

Measurements. The simulation model in this experiment was run thirty-two times using all possible combinations of input variable levels. This was accomplished by using different input files for each run for the variable factors, and common input files for the fixed factors. Examples of these files are shown in Appendix D: Input Files SIMU7 and SIMU9, and Tables I through VI. SIMU7 is an example of one of the combinations of input variable levels depicted in Table VII: Variable Factors. Table VIII: Simulation Output, shows the response value of the dependent variable observed for each of the thirty-two treatment combinations run on the model. The designators A0/A1, B0/B1, C0/C1, D0/D1, and E0/E1 reflect the low and high levels, respectively, of the independent variables shown in Table VII.

Method. These results were evaluated by ANOVA using the Yates' Method of computing factorial effects totals. This method is the "most expeditious" computational method for looking at multiple factorial effects (3:158). The effect means calculated by the Yates' Method are shown in Table IX. The lower case letters (a-e) in the left hand column of Table IX indicate which variables are at a high level in that treatment. If a letter is not present in a treatment its corresponding variable was set low for that run. All calculations were performed with the full number of significant digits allowed on the CDC CYBER computer system. The calculations shown in Table IX, except for the treatment results and effect means, have been rounded to whole numbers for readability.

TABLE IX

Effect Means

TREATMENT	RESULT	(1)	(2)	(3)	(4)	EFFECT TOTAL	EFFECT MEAN
(1)	0.00	0	20	41	186	477	29.83
a	0.00	20	21	145	292	-1	-.04
b	10.34	0	71	80	-3	477	29.79
ab	9.68	21	73	211	2	-0	-.01
c	0.00	0	40	-1	185	5	.32
ac	0.00	71	41	-1	291	0	.02
bc	10.94	0	105	3	-2	5	.29
abc	10.27	73	106	-1	2	1	.04
d	0.03	0	-1	41	3	234	14.63
ad	0.00	40	-1	144	2	-4	-.22
bd	36.11	0	-1	80	0	233	14.58
abd	35.26	41	-1	211	-0	-3	-.19
cd	0.23	0	2	-1	3	1	.09
acd	0.01	105	1	-1	2	2	.10
bcd	36.62	0	-1	3	1	1	.06
abcd	36.28	106	-0	-1	-0	2	.12
e	0.00	0	20	1	103	106	6.61
ae	0.00	-1	21	2	131	5	.31
be	19.09	0	71	1	-0	106	6.64
abe	20.79	-1	73	2	-3	4	.28
ce	0.00	-0	40	-0	103	-1	-.05
ace	0.00	-1	41	0	131	-0	-.02
bce	19.74	-0	105	-1	0	-0	-.02
abce	20.79	-0	106	1	-3	-1	-.05
de	0.02	0	-1	1	1	27	1.71
ade	0.01	2	-1	1	1	-3	-.21
bde	52.70	0	-1	1	0	28	1.74
abde	52.09	1	-0	1	1	-4	-.23
cde	0.03	-0	2	-0	0	0	.02
acde	0.02	-1	1	1	1	1	.06
bcde	53.15	-0	-1	-1	1	1	.04
abcde	53.14	-0	0	1	1	1	.03

Analysis

The sum of squares for total and each treatment, which were calculated from the effect totals in Table IX, are shown in Table X: Sum of Squares.

TABLE X

SUM OF SQUARES			
SOURCE	DF	SS	MS
(1)	1	7121	7121
A	1	0	0
B	1	7100	7100
AB	1	0	0
C	1	1	1
AC	1	0	0
BC	1	1	1
ABC	1	0	0
D	1	1712	1712
AD	1	0	0
BD	1	1702	1702
ABD	1	0	0
CD	1	0	0
ACD	1	0	0
BCD	1	0	0
ABCD	1	0	0
E	1	350	350
AE	1	1	1
BE	1	352	352
ABE	1	1	1
CE	1	0	0
ACE	1	0	0
BCE	1	0	0
ABCE	1	0	0
DE	1	24	24
ADE	1	0	0
BDE	1	24	24
ABDE	1	0	0
CDE	1	0	0
ACDE	1	0	0
BCDE	1	0	0
ABCDE	1	0	0
TOTALS	31	11268	

An examination of this data, again rounded for readability, revealed negligible effects for the third through fifth order-interactions. Thus the treatment sum of squares was calculated on the main effects and second-order interactions only. This gave fifteen degrees of freedom for the treatments sum of squares and sixteen degrees of freedom for the error sum of squares (see Table XI: ANOVA).

Table XI

ANOVA

SOURCE	D.F	S.S.	M.S.
TREATMENTS	15	4121.09	274.74
ERROR	16	7146.88	446.68
TOTAL	31	11267.97	

TABLE XII

COMPARISON VALUES

STANDARD ERROR	=	7.47
T VALUE AT .10	=	1.75
T VALUE AT .05	=	2.12
T VALUE AT .01	=	2.92
COMPARISON VALUE AT .10	=	13.05
COMPARISON VALUE AT .05	=	15.84
COMPARSION VALUE AT .01	=	21.83

The error mean square of 446.68 was used to compute the standard error of 7.47. This standard error was multiplied by the critical values of t (for 16 degrees of freedom) at the alpha = .01, .05, and .10 levels of significance (shown in Table XII), and resulted in comparison values of 13.05 at alpha = .10, 15.84 at alpha = .05, and 23.83 at alpha = .01.

Significance of Results

The effect means for treatments (TABLE IX) were contrasted with the comparison values (TABLE XII) to determine the significance of the effects of each treatment. If the effect mean exceeded a given comparison value the main effect (or interaction effect) for that treatment is statistically significant at the corresponding alpha level. Main effect B, transportation time, was significant at the alpha = .01 level. Main effect D, maintenance duration, as well as the BD interaction, were significant at the alpha = .10 level.

IV. Summary, Conclusions, and Recommendations

The six objectives this research was to meet were:

1. To conduct a thorough review of the ALCM spare engine system to determine the system's relevant variables and their interrelationships, and the amount and type of variability in these relationships.
2. To build a computer simulation model of the system that would reflect the findings of this review.
3. To experiment with the model and assess the significance of changes in the variables on the system's operational capability.
4. To document the derivation, use, applicability, and significance of the model's output.
5. To incorporate flexibility in the model's design to allow for changes that will occur in the system's future.
6. To answer three research questions:
   a. Do any of the selected variables have a significant effect on the ALCM engine operational capability when allowed to vary within their probable range of values ?
   b. Which of these variables are significant ?
   c. What are the implications of these findings ?

This chapter will summarize the effort of the authors to meet these objectives, and will offer recommendations for further study in this area.

Summary

The logistics issue addressed in the research is the management of spares acquisition and logistics support activites for the Air Launched Cruise Missile engine. Effective management of a complex weapon system such as the ALCM requires careful consideration of the overall logistics support given to the system. This consideration includes the number of repairable spare parts to purchase, the timing and frequency of maintenance for these parts, and the effect of both of these aspects on the operational capability of the weapon system, throughout its useful life. This research focuses on an alternative method, from those currently employed by the Air Force, for analyzing the spare engine procurement and support requirements of the ALCM.

Objective 1. The ALCM system was thoroughly reviewed by the authors to determine the environment and characteristics of the system. Support for this review was provided by Engine Logistics Office of the Aeronautical Systems Division at Wright-Patterson AFB, Ohio. A conceptual model of the system was developed from the information obtained from this office. The model was revised and corrected, then reviewed by the Engine Logistics Office, which confirmed that the model was an accurate representation of the real-world ALCM system.

Objective 2. A computer model was then written in the SIMSCRIPT II.5 language. This model was designed as a tool that ALCM system managers could use to understand the ALCM system and to determine the impact that changes in selected variables might have on the system's performance.

The authors selected five variables that appeared to have the potential for significant impact on the ALCM system. The possible range of values for these variables was then determined by reviewing the available information on the system.

<u>Objective 3</u>. An experimental design was then selected that would determine the actual significance of changes in these variables on the model's dependent variable. The factorial design chosen allowed the determination of the impact of both the main and interaction effects with a single analysis. The experiment was conducted on the CDC CYBER computer and analyzed using the Yates' method for ANOVA. It was found that the transportation time was statistically significant at the alpha = .01 level, and the maintenance duration and the transportation/maintenance interaction was significant at the alpha = .10 level.

<u>Objective 4</u>. Appendices A through G document the computer model's operation, verification, input requirements, and output. Appendix A illustrates some of the capabilities of the model to provide estimates of future states of the ALCM system that managers may find valuable for long range system planning. Appendix B explains some of the terms peculiar to the SIMSCRIPT language that are used in the model. Appendix C explains, in detail, the line by line

operation and design logic of the model. This appendix will give other modelers the depth of understanding necessary to adapt and modify the model for their particular use. Appendix D gives examples of the input files required to run the model and specifies the values of those factors considered as constants in this research. Appendix E is the numbered program listing referenced in the other appendices. Appendix F is an explanation of the terms and variables used in the model. Appendix G is an example of the procedure used to verify the correct operation of the model.

Objective 5. The computer model was designed with the flexibility to easily modify all the assumptions and limitations built into it, so that newly discovered relevant variables and different levels of factors could be incorporated as the system evolves. In most cases this can be accomplished by modifying a short input file found in Appendix D. The documentation found in the other appendices will give an experienced modeler the flexibility to explore numerous alternative courses of action in experimenting with this model.

Conclusions

Objective 6a. The analysis of the experimental results, presented in Chapter 3, clearly shows that some of the selected variables do have a significant impact on the system's operational capability, as defined by the authors.

Objective 6b. Transportation time, maintenance duration, and their interaction proved significant at the alpha = .10 level. This conclusion is based on the mean effects and comparison values depicted in Tables IX and XII of Chapter 3. All the other variables in the study were not significant below the alpha = .40 level.

Recommendations

Objective 6c. The first recommendation presented is also the most important. It is the necessity for full validation of the model. Without this validation, the significant effects shown for transportation time and maintenance duration should not be acted on by ALCM system managers. Validation is necessary to confirm the assumptions inherent in the models underlying structure and constant parameter values before the results of this research can be accepted as accurate.

This validation should include an investigation of the assumptions made in the model. First, depot 2 (OCALC) appears to be under-utilized in its first four years of operation because of the policy that limits depot 2 to performing only limited overhauls (20). Second, the assumption, made by the authors, that selection of a spare engine from a particular depot is made on the basis of the depot with the most spare engines available, may be invalid. A policy of selecting the oldest engine from either depot's stocks may be more appropriate. Third, the depot selected to service a particular engine requiring a limited overhaul was based on using the depot with the lowest utilization rate, whereas a regional servicing policy may be more appropriate. Fourth, the author's use of the model's engine conversion routine is based on an attempt to convert as many engines as possible during their longer servicing period. The emphasis here is on conserving maintenance resources by performing extensive maintenance (the conversion) only on an engine that would normally be scheduled for extended maintenance (full, repair, and refurbishment services). Other considerations may be more important than this emphasis in determining the conversion schedule.

The next recommendation is an in-depth study of the utility of the dependent variable, as expressed in the model, as an indicator of the ALCM engine's operational capability. The actual correlation between the amount of time an engine spends as an operational asset beyond its warranted life, and its probability of successfully firing and operating when a demand is placed on it, has not yet been established. Further analysis of the results of the current test program, and an extension of this program to include engines operational past their warranted life, may be required to establish this correlation. Once this correlation has been established, a new experiment should be performed to test for a significant change in system capability due to changes in the transportation time and maintenance duration parameters. Even though these parameters, as used, currently indicate a significant effect on the dependent variable, the dependent variable's actual correlation with system capability (possibly at less than a one to one ratio) may not indicate a significant change in this capability.

Finally, the assumption of a uniform distribution for each of the stochastic processes in the model is based on inadequate information concerning the actual probability distributions that should apply in each case. These distributions should be determined from an analysis of engineering data and test results, and the model updated to reflect the correct distributions.

Once the verification is complete and the assumptions, as presented, are verified, an in-depth study of transportation time and maintenance duration should be conducted to reveal the actual distribution of values these variables are likely to assume. The authors recommend incorporating accurate probability distribution functions for each of these significant variables into the present model, then experimenting with the model to produce accurate predictions of the dependent variable and the other output variables mentioned in Appendix A.

Furthermore, once the model is validated and accurate probability distribution functions for the significant variables are incorporated, the model may be used to test the effects of changes in management policy. For example, the model has an untested capability to vary the engine servicing capacity of each depot at any time. This capability, in conjunction with the (untested) capability to allow the early (before the due date) overhaul of an engine, may significantly affect the operation of the model. Management may adopt a policy of early overhauls to smooth out the "peaks and valleys" in service requirements, and to insure that these "peaks" do not exceed a less-than-unlimited service capacity at a depot. The model will be able to show management the implications of this change in policy.

A final recommendation is to examine the possibility of adapting the ALCM model to reflect the operational environment of the Ground Launched Cruise Missile (GLCM) or the Submarine Launched Cruise Missile (SLCM), both of which use the same type of engine as the ALCM. These systems are further behind the ALCM in development, and final decisions on the number of spare engines to procure have not been made. A slightly modified model could be used to determine the effects on the system of different numbers of spare engines. An evaluation of these effects could help managers determine the appropriate number of spares to procure.

Appendix A: Simulation Output Summary

This appendix illustrates some of the capabilities of the model to provide estimates of future states of the ALCM system that managers may find valuable for long range system planning. It is divided into three parts, the first giving a "snapshot" of the state of the system at 0000 hours on the first day of each month (pages 54-60). The second part summarizes the activities that have occurred for each month of the simulation run (pages 61-67). The third part reviews the system's operation over its entire simulated life, records the total number of broken and overdue missiles, and their average time on base past their failure or overdue dates (page 67). This last item is the dependent variable examined in this research effort.

The following is an explanation of the codes used in part 1 of the output summary (the numbers [1-12] down the left column, under each year, indicate the month for each row of output):

NQ1 and NQ2 - the number of engines waiting for servicing at each depot because of a shortage in the depot's service capability (depot 1 refers to Williams and depot 2 refers to OCALC).

NX1 and NX2 - the number of engines being serviced by each depot.

NR1 and NR2 - the number of repaired, serviceable engines in each depot's stocks.

NTS - the number of engines currently being tested.

NAS - the total number of operational engines deployed to bases in the system.

NSS - the number of spare engines required by the system to replace all broken and overdue engines, provide for spare test engines, and fill the transportation and maintenance pipeline. This number does not count engines in depot repaired stocks if there is no demand for them.

NKS - the number of engines that require a spare engine, due to failure or being overdue for scheduled maintenance, but cannot obtain one due to lack of their availability. This number indicates the excess of demand over supply for spare engines.

NBN - the number of failed engines awaiting a replacement spare engine. A broken engine will stay on base a minimum of TRANSPORT.DAYS before being replaced due to the demand-pull supply concept used in the system.

NOV - the number of engines in a deployed status that have exceeded their warranted lifetimes since their last overhaul. Due to the structure of the model, these engine are counted as being on base a minimum of TRANSPORT.DAYS before being replaced. However, the spare engine reorder process is actually initiated TRANSPORT.DAYS prior to the engine's actual overdue date. An adjustment is made in the model's output to correct these excess overdue days.

A knowledge of the most probable state of these variables can be extremely valuable to system managers in determining depot maintenance requirements over the years ahead. The NAS counter can give a clear picture of operational missile strength as it might change over the system's life. Combined with the information available from the main output variable, the average time a missile exceeds it warranted life while deployed, system managers should be able to determine probable readiness levels for the ALCM system throughout its useful life.

Trends shown in the "snapshot" pictures (part 1) of the system's variables can give managers a great deal of useful information. For instance, in the output example shown in this appendix, the system seems capable of handling the demand for spare engine for all years except 1987 to 1992. The demand quickly exceeds the supply and stays that way for five years, with demand peaking at 407 engines and only 115 spare engines available. This trend should indicate that further research into the cause of this apparent "bottle-neck" may be worthwhile so that solutions can be developed in time to prevent the problem from occurring. This may require the use of premium transportation for spare engines, or expediting depot operations. The model can indicate where and when problem areas may occur and what their most likely causes are so that action can be taken early enough to prevent these problems.

Part 2 of the output summary gives additional information that will interest system managers, especially depot managers. It gives monthly information on the following variables:

INCR - the increase in the number of spare engines required over the previous month.

TREQ - the maximum number of engines required at any time during the month.

A101,A104 - the average age (in days past the engine's last overhaul date) for each type (101 or 104) of engine. These counters apply only to deployed engines. This may serve as another indicator of the readiness of the ALCM fleet.

MINR - the number of limited overhauls performed by the depots during the month.

MAJR - the number of full overhauls performed by the depots during the month.

UNSC - the number of unscheduled overhauls (due to failure) performed by the depots during the month.

REFB - the number of test refurbishments performed by the depots during the month.

CONV - the number of 101 to 104 conversions performed by the depots during the month.

Part 3 of the output summary shows the total number of engines that have failed or exceeded their warranted life before being serviced. Since there are only 1830 engines manufactured and failures or overdues may exceed 7000, these figures reflect the many "lifetimes" of each engine over the life of the simulation. An individual engine may have failed early for three out of the five times it was deployed or redeployed to an operational base. This will be reflected as three failed engines in the system's counter.

Because of the model's structure, all engines that are exchanged at or past their warranted life are counted as overdue (those that meet or exceed their warranted life). The total number overdue will thus reflect the total number of engines deployed that do not fail and are not used for tests. The "average number of days overdue" thus reflects the fleet-wide average days overdue for those engines that have met or exceeded their warranted life.

PART 1

	N	N	N	N	N	N	N	N	N	N	N	N
	Q	Q	X	X	R	R	T	A	S	K	B	O
	1	2	1	2	1	2	S	S	S	S	N	V
1981												
1	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0	0	0	0
12	0	0	0	0	0	0	0	0	0	0	0	0

	N	N	N	N	N	N	N	N	N	N	N	N
	Q	Q	X	X	R	R	T	A	S	K	B	O
	1	2	1	2	1	2	S	S	S	S	N	V
1982												
1	0	0	0	0	0	0	0	5	0	0	0	0
2	0	0	0	0	5	0	0	10	0	0	0	0
3	0	0	0	0	10	0	0	20	0	0	0	0
4	0	0	0	0	15	0	0	32	0	0	0	0
5	0	0	0	0	20	0	0	41	0	0	0	0
6	0	0	0	0	25	0	0	62	0	0	0	0
7	0	0	0	0	30	0	0	82	0	0	0	0
8	0	0	0	0	34	0	1	110	1	0	0	0
9	0	0	0	0	39	0	2	141	1	0	0	0
10	0	0	1	0	44	0	1	173	2	0	0	0
11	0	0	2	0	49	0	0	214	2	0	0	0
12	0	0	1	0	54	0	0	245	1	0	0	0

	N	N	N	N	N	N	N	N	N	N	N	N
	Q	Q	X	X	R	R	T	A	S	K	B	O
	1	2	1	2	1	2	S	S	S	S	N	V
1983												
1	0	0	0	0	60	0	0	295	0	0	0	0
2	0	0	0	0	65	0	1	333	0	0	0	0
3	0	0	0	0	69	0	2	363	1	0	0	0
4	0	0	1	0	73	0	2	411	3	0	0	0
5	0	0	1	0	78	0	2	451	3	0	0	0
6	0	0	2	0	82	0	2	490	4	0	0	0
7	0	0	3	0	86	0	3	530	5	0	0	0
8	0	0	2	0	91	0	2	570	4	0	0	0
9	0	0	2	0	96	0	2	609	5	0	0	0
10	0	0	2	0	102	0	3	649	4	0	0	0
11	0	0	1	0	107	0	3	688	3	0	0	0

12	0	0	2	0	111	0	2	716	4	0	0	0

	N	N	N	N	N	N	N	N	N	N	N	N
	Q	Q	X	X	R	R	T	A	S	K	B	O
	1	2	1	2	1	2	S	S	S	S	N	V
1984												
1	0	0	2	0	111	0	1	762	5	0	0	0
2	0	0	2	0	111	0	3	797	5	0	0	0
3	0	0	2	0	111	0	5	833	5	0	0	0
4	0	0	7	0	102	0	3	868	14	0	0	2
5	0	0	15	0	94	0	2	905	23	0	0	3
6	0	0	14	0	93	0	2	941	23	0	0	2
7	0	0	15	0	87	0	2	976	28	0	0	5
8	0	0	24	0	76	0	3	1013	39	0	0	6
9	0	0	25	0	70	0	4	1047	45	0	0	8
10	0	0	36	0	61	0	3	1082	55	0	0	9
11	0	0	38	0	59	0	3	1120	58	0	0	10
12	0	0	38	0	55	0	3	1151	61	0	0	12

	N	N	N	N	N	N	N	N	N	N	N	N
	Q	Q	X	X	R	R	T	A	S	K	B	O
	1	2	1	2	1	2	S	S	S	S	N	V
1985												
1	0	0	44	0	49	0	2	1177	67	0	0	8
2	0	0	46	0	47	0	3	1206	70	0	0	10
3	0	0	43	0	50	0	4	1234	66	0	0	9
4	0	0	44	0	49	0	3	1261	67	0	0	9
5	0	0	45	0	48	0	3	1289	68	0	0	9
6	0	0	45	0	47	0	3	1314	68	0	0	10
7	0	0	48	0	47	0	2	1340	69	0	0	7
8	0	0	43	0	52	0	3	1366	65	0	0	8
9	0	0	43	0	46	0	3	1392	71	0	0	11
10	0	0	50	0	44	0	4	1417	72	0	0	12
11	0	0	45	0	48	0	3	1445	69	0	0	11
12	0	0	48	0	47	0	3	1464	70	0	0	10

	N	N	N	N	N	N	N	N	N	N	N	N
	Q	Q	X	X	R	R	T	A	S	K	B	O
	1	2	1	2	1	2	S	S	S	S	N	V
1986												
1	0	0	47	0	46	0	3	1488	70	0	0	11
2	0	0	45	0	48	0	2	1508	69	0	0	10
3	0	0	46	0	47	0	3	1528	70	0	0	9
4	0	0	47	0	47	0	4	1547	69	0	0	10
5	0	0	46	0	47	0	3	1567	70	0	0	10
6	0	0	45	0	48	0	3	1587	69	0	0	9
7	0	0	45	0	52	0	3	1601	64	0	0	8
8	0	0	39	0	55	0	2	1626	62	0	0	9
9	0	0	44	0	43	0	3	1646	74	0	0	11
10	0	0	52	0	47	0	4	1665	69	0	0	5
11	0	0	33	0	63	0	3	1685	54	0	0	11
12	0	0	48	0	47	0	3	1690	70	0	0	8

1987

	NQ1	NQ2	NX1	NX2	NR1	NR2	NTS	NAS	NSS	NKS	NBN	NOV
1	0	0	52	0	43	0	4	1689	73	0	0	9
2	0	0	54	0	39	0	3	1687	77	0	0	10
3	0	0	58	0	30	0	4	1688	87	0	0	11
4	0	0	76	0	15	0	4	1687	102	0	0	11
5	0	0	85	0	2	0	3	1686	115	9	0	14
6	0	0	86	0	0	0	3	1687	124	20	0	20
7	0	0	81	0	0	0	4	1687	132	31	0	31
8	0	0	86	0	0	0	2	1686	153	46	0	46
9	0	0	87	0	0	0	2	1686	170	67	0	67
10	0	0	85	0	0	0	4	1685	186	80	0	80
11	0	0	86	0	0	0	3	1684	203	96	0	96
12	0	0	86	0	0	0	3	1685	213	107	0	07

1988

	NQ1	NQ2	NX1	NX2	NR1	NR2	NTS	NAS	NSS	NKS	NBN	NOV
1	0	0	83	0	0	0	3	1684	218	112	0	12
2	0	0	91	0	0	0	2	1684	232	122	0	22
3	0	0	87	0	0	0	3	1684	241	132	0	32
4	0	0	84	0	0	0	4	1684	251	146	0	46
5	0	0	87	0	0	0	3	1682	268	158	0	58
6	0	0	85	0	0	0	3	1683	277	169	0	69
7	0	0	88	0	0	0	4	1682	292	184	0	84
8	0	0	90	0	0	0	4	1682	305	194	0	94
9	0	0	96	0	0	0	2	1681	336	221	0	21
10	0	0	81	0	0	0	2	1682	347	240	0	40
11	0	0	96	0	0	0	4	1681	372	254	0	54
12	0	0	90	0	0	0	4	1680	375	263	0	63

1989

	NQ1	NQ2	NX1	NX2	NR1	NR2	NTS	NAS	NSS	NKS	NBN	NOV
1	0	0	92	0	0	0	3	1679	367	252	0	52
2	0	0	94	0	0	0	3	1680	369	256	0	56
3	0	0	84	0	0	0	3	1680	358	248	0	48
4	0	0	84	10	0	0	3	1679	362	245	0	45
5	0	0	69	10	0	0	3	1678	358	250	0	50
6	0	0	94	4	0	0	3	1679	332	213	0	13
7	0	0	94	0	0	0	3	1679	339	225	0	25
8	0	0	80	0	0	0	4	1678	314	201	0	01
9	0	0	102	0	0	0	4	1677	323	199	0	99
10	0	0	80	0	0	0	2	1678	312	202	0	02
11	0	0	96	0	0	0	2	1678	317	197	0	97
12	0	0	92	0	0	0	4	1677	334	220	0	20

	N	N	N	N	N	N	N	N	N	N	N	N
	Q	Q	X	X	R	R	T	A	S	K	B	O
	1	2	1	2	1	2	S	S	S	S	N	V
1990												
1	0	0	83	0	0	0	4	1675	331	213	0	13
2	0	0	101	0	0	0	3	1675	354	230	0	30
3	0	0	82	0	0	0	2	1676	357	245	0	45
4	0	0	98	0	0	0	2	1675	359	238	0	38
5	0	0	92	0	0	0	3	1674	380	264	0	64
6	0	0	78	0	0	0	3	1674	368	252	0	52
7	0	0	101	0	0	0	2	1674	397	273	0	73
8	0	0	80	0	0	0	3	1673	397	285	0	85
9	0	0	98	0	0	0	4	1672	391	266	0	66
10	0	0	93	0	0	0	3	1672	389	272	0	72
11	0	0	80	0	0	0	2	1673	339	223	0	23
12	0	0	97	0	0	0	3	1673	328	204	0	04

	N	N	N	N	N	N	N	N	N	N	N	N
	Q	Q	X	X	R	R	T	A	S	K	B	O
	1	2	1	2	1	2	S	S	S	S	N	V
1991												
1	0	0	77	0	0	0	4	1671	289	177	0	77
2	0	0	96	0	0	0	4	1671	257	133	0	33
3	0	0	93	0	0	0	3	1672	227	109	0	09
4	0	0	78	0	0	0	4	1672	162	46	0	46
5	0	0	97	0	0	0	3	1670	156	32	0	32
6	0	0	77	0	0	0	3	1670	138	25	0	25
7	0	0	92	0	5	0	3	1671	125	7	0	6
8	0	0	84	0	13	0	3	1671	116	6	0	6
9	0	0	58	0	36	0	4	1669	92	7	0	7
10	0	0	63	0	38	0	4	1669	91	5	0	5
11	0	0	60	0	23	0	3	1669	106	13	0	13
12	0	0	75	0	27	0	2	1670	102	1	0	1

	N	N	N	N	N	N	N	N	N	N	N	N
	Q	Q	X	X	R	R	T	A	S	K	B	O
	1	2	1	2	1	2	S	S	S	S	N	V
1992												
1	0	0	57	0	44	0	2	1668	84	0	0	0
2	0	0	37	0	63	0	3	1669	66	0	0	0
3	0	0	40	0	69	0	4	1669	59	0	0	0
4	0	0	21	0	75	0	3	1669	53	0	0	0
5	0	0	38	0	65	0	3	1668	64	0	0	0
6	0	0	35	0	74	0	3	1668	54	0	0	0
7	0	0	32	0	58	0	2	1667	70	0	0	7
8	0	0	59	0	42	0	3	1668	87	0	0	5
9	0	0	62	0	38	0	4	1667	90	0	0	0
10	0	0	42	0	65	0	3	1666	63	0	0	0
11	0	0	15	0	92	0	3	1667	36	0	0	3
12	0	0	24	0	73	0	4	1666	54	0	0	5

	N	N	N	N	N	N	N	N	N	N	N	N
	Q	Q	X	X	R	R	T	A	S	K	B	O
	1	2	1	2	1	2	S	S	S	S	N	V
1993												
1	0	0	40	0	68	0	2	1664	59	0	0	0
2	0	0	22	0	87	0	2	1665	41	0	0	0
3	0	0	6	0	92	0	3	1665	35	0	0	8
4	0	0	9	18	76	0	3	1665	51	0	0	6
5	0	0	18	12	38	18	3	1664	71	0	0	8
6	0	0	24	18	22	29	3	1665	76	0	0	7
7	0	0	15	26	28	23	2	1664	76	0	0	8
8	0	0	28	8	26	29	2	1665	72	0	0	9
9	0	0	14	22	26	28	3	1663	73	0	0	10
10	0	0	18	31	17	17	3	1664	93	0	0	10
11	0	0	28	9	24	44	3	1663	59	0	0	3
12	0	0	3	10	44	44	3	1664	39	0	0	4

	N	N	N	N	N	N	N	N	N	N	N	N
	Q	Q	X	X	R	R	T	A	S	K	B	O
	1	2	1	2	1	2	S	S	S	S	N	V
1994												
1	0	0	14	5	35	36	2	1662	57	0	0	9
2	0	0	24	10	34	34	2	1663	59	0	0	3
3	0	0	23	15	30	32	2	1663	65	0	0	3
4	0	0	12	7	41	43	2	1664	43	0	0	3
5	0	0	4	10	45	42	2	1662	40	0	0	4
6	0	0	13	0	42	47	3	1663	37	0	0	6
7	0	0	9	11	36	37	3	1663	53	0	0	13
8	0	0	21	16	31	32	1	1663	64	0	0	8
9	0	0	16	12	36	37	1	1661	54	0	0	7
10	0	0	1	27	30	29	3	1662	67	0	0	17
11	0	0	30	0	25	44	3	1662	57	0	0	8
12	0	0	8	15	31	31	2	1662	65	0	0	14

	N	N	N	N	N	N	N	N	N	N	N	N
	Q	Q	X	X	R	R	T	A	S	K	B	O
	1	2	1	2	1	2	S	S	S	S	N	V
1995												
1	0	0	22	17	31	39	2	1660	57	0	0	0
2	0	0	3	1	48	55	3	1661	23	0	0	1
3	0	0	4	8	38	39	2	1661	49	0	0	15
4	0	0	20	25	28	26	1	1661	73	0	0	5
5	0	0	16	10	31	29	2	1659	67	0	0	10
6	0	0	25	18	24	30	3	1660	71	0	0	4
7	0	0	7	10	41	39	3	1660	45	0	0	7
8	0	0	17	24	16	17	2	1660	93	0	0	18
9	0	0	42	11	17	27	1	1658	82	0	0	8
10	0	0	5	18	27	28	2	1659	69	0	0	13
11	0	0	26	39	13	13	3	1659	98	0	0	7
12	0	0	29	4	22	40	2	1659	62	0	0	6

	N	N	N	N	N	N	N	N	N	N	N	N
	Q	Q	X	X	R	R	T	A	S	K	B	O
	1	2	1	2	1	2	S	S	S	S	N	V
1996												
1	0	0	13	30	14	15	1	1657	94	0	0	14
2	0	0	30	24	21	16	2	1658	85	0	0	7
3	0	0	16	10	23	26	3	1658	72	0	0	14
4	0	0	27	40	9	9	1	1658	104	0	0	7
5	0	0	23	13	28	30	1	1656	64	0	0	5
6	0	0	32	9	14	15	3	1657	92	0	0	16
7	0	0	10	46	37	9	1	1657	76	0	0	0
8	0	0	3	1	35	34	1	1657	53	0	0	15
9	0	0	33	28	11	15	3	1656	96	0	0	6
10	0	0	9	22	29	31	2	1657	61	0	0	6
11	0	0	9	20	29	27	1	1657	66	0	0	8
12	0	0	28	6	26	34	2	1657	62	0	0	2

	N	N	N	N	N	N	N	N	N	N	N	N
	Q	Q	X	X	R	R	T	A	S	K	B	O
	1	2	1	2	1	2	S	S	S	S	N	V
1997												
1	0	0	14	15	25	25	1	1657	71	0	0	12
2	0	0	12	35	26	15	1	1658	79	0	0	5
3	0	0	15	4	35	38	1	1658	47	0	0	9
4	0	0	2	11	31	30	0	1658	58	0	0	10
5	0	0	29	2	31	40	0	1658	48	0	0	1
6	0	0	1	2	39	40	0	1658	40	0	0	12
7	0	0	17	9	30	30	0	1658	59	0	0	6
8	0	0	0	15	44	41	0	1658	33	0	0	1
9	0	0	2	8	31	31	0	1658	55	0	0	11
10	0	0	27	16	20	26	0	1658	70	0	0	4
11	0	0	6	21	29	27	0	1658	60	0	0	8
12	0	0	19	1	34	44	0	1658	38	0	0	3

	N	N	N	N	N	N	N	N	N	N	N	N
	Q	Q	X	X	R	R	T	A	S	K	B	O
	1	2	1	2	1	2	S	S	S	S	N	V
1998												
1	0	0	1	3	50	45	0	1658	21	0	0	0
2	0	0	0	11	36	36	0	1658	43	0	0	7
3	0	0	17	9	30	37	0	1658	48	0	0	2
4	0	0	0	6	45	45	0	1658	24	0	0	1
5	0	0	18	0	36	36	0	1658	42	0	0	2
6	0	0	34	7	19	19	0	1658	76	0	0	7
7	0	0	52	1	12	11	0	1658	91	0	0	12
8	0	0	73	8	2	1	0	1658	111	0	0	6
9	0	0	69	0	7	6	0	1658	101	0	0	7
10	0	0	67	2	6	4	0	1658	104	0	0	9
11	0	0	72	1	0	0	0	1658	120	17	0	17
12	0	0	79	2	4	0	0	1658	110	1	0	1

	N	N	N	N	N	N	N	N	N	N	N	N
	Q	Q	X	X	R	R	T	A	S	K	B	O
	1	2	1	2	1	2	S	S	S	S	N	V
1999												
1	0	0	55	1	33	2	0	1658	79	3	0	3
2	0	0	34	2	53	3	0	1658	58	3	0	3
3	0	0	27	0	56	5	0	1658	53	6	0	6
4	0	0	22	19	36	5	0	1658	73	5	0	5
5	0	0	30	5	31	24	0	1658	59	3	0	3
6	0	0	30	2	30	29	0	1658	55	0	0	2
7	0	0	28	1	38	31	0	1658	45	0	0	0
8	0	0	27	1	30	31	0	1658	53	0	0	2
9	0	0	52	1	25	19	0	1658	70	0	0	1
10	0	0	58	1	15	14	0	1658	85	0	0	0
11	0	0	46	1	29	11	0	1658	74	0	0	4
12	0	0	63	1	26	8	0	1658	80	0	0	0

	N	N	N	N	N	N	N	N	N	N	N	N
	Q	Q	X	X	R	R	T	A	S	K	B	O
	1	2	1	2	1	2	S	S	S	S	N	V
2000												
1	0	0	45	1	25	9	0	1658	80	0	0	10
2	0	0	56	2	33	7	0	1658	74	0	0	0
3	0	0	44	1	42	8	0	1658	63	0	0	0
4	0	0	8	0	70	9	0	1658	33	0	0	3
5	0	0	48	1	25	9	0	1658	78	0	0	4
6	0	0	65	2	4	5	0	1658	103	0	0	15
7	0	0	63	1	17	0	0	1658	94	5	0	5
8	0	0	58	1	29	1	0	1658	81	2	0	2
9	0	0	49	1	12	2	0	1658	96	14	0	14
10	0	0	88	1	0	0	0	1658	112	5	0	5
11	0	0	70	1	0	0	0	1658	117	22	0	22
12	0	0	66	1	0	0	0	1658	114	15	0	15

PART 2

```
THE MAXIMUM NUMBER OF SPARE ENGINES REQUIRED OVER THE
LIFE OF THE SYSTEM IS  407


 1981  INCR   TREQ   A101   A104   MINR   MAJR   UNSC   REFB   CONV
  JAN   0      0      0      0      0      0      0      0      0
  FEB   0      0      0      0      0      0      0      0      0
  MAR   0      0      0      0      0      0      0      0      0
  APR   0      0      0      0      0      0      0      0      0
  MAY   0      0      0      0      0      0      0      0      0
  JUN   0      0      0      0      0      0      0      0      0
  JUL   0      0      0      0      0      0      0      0      0
  AUG   0      0      0      0      0      0      0      0      0
  SEP   0      0      0      0      0      0      0      0      0
  OCT   0      0      0      0      0      0      0      0      0
  NOV   0      0      0      0      0      0      0      0      0
  DEC   0      0      0      0      0      0      0      0      0

 1982  INCR   TREQ   A101   A104   MINR   MAJR   UNSC   REFB   CONV
  JAN   0      0     199     0      0      0      0      0      0
  FEB   0      0     218     0      0      0      0      0      0
  MAR   0      0     226     0      0      0      0      0      0
  APR   0      0     236     0      0      0      0      0      0
  MAY   0      0     252     0      0      0      0      0      0
  JUN   0      0     255     0      0      0      0      0      0
  JUL   0      0     259     0      0      0      0      0      0
  AUG   1      1     263     0      0      0      0      0      0
  SEP   0      0     267     0      0      0      0      1      0
  OCT   1      2     275     0      0      0      0      1      0
  NOV   0      0     280     0      0      0      0      0      0
  DEC   0      1     292     0      0      0      0      0      0

 1983  INCR   TREQ   A101   A104   MINR   MAJR   UNSC   REFB   CONV
  JAN   0      0     298     0      0      0      0      0      0
  FEB   0      0     308     0      0      0      0      0      0
  MAR   0      1     322     0      0      0      0      1      0
  APR   1      3     329     0      0      0      0      0      0
  MAY   0      0     342     0      0      0      0      2      0
  JUN   1      4     354     0      0      0      0      1      0
  JUL   1      5     367     0      0      0      0      1      0
  AUG   0      5     382     0      0      0      0      1      0
  SEP   0      5     395     0      0      0      0      1      0
  OCT   0      4     409     0      0      0      0      1      0
  NOV   0      5     422     0      0      0      0      1      0
  DEC   0      4     441     0      0      0      0      1      0
```

1984	INCR	TREQ	A101	A104	MINR	MAJR	UNSC	REFB	CONV
JAN	0	5	455	0	0	0	0	1	0
FEB	1	6	470	0	0	0	0	1	0
MAR	0	6	484	0	3	0	0	3	0
APR	8	14	498	0	10	0	0	2	0
MAY	9	23	511	0	13	0	0	0	0
JUN	2	25	523	0	14	0	0	1	0
JUL	5	30	534	0	23	0	0	1	0
AUG	10	40	541	0	25	0	0	1	0
SEP	7	47	543	0	33	0	0	2	0
OCT	8	55	538	0	36	0	0	2	0
NOV	5	60	529	0	36	0	0	0	0
DEC	3	63	521	0	43	0	0	2	0

1985	INCR	TREQ	A101	A104	MINR	MAJR	UNSC	REFB	CONV
JAN	7	70	509	0	44	0	0	2	0
FEB	2	72	500	0	38	0	0	1	0
MAR	0	71	491	0	44	0	0	2	0
APR	0	72	484	0	42	0	0	1	0
MAY	1	73	477	0	45	0	0	1	0
JUN	0	71	472	0	46	0	0	1	0
JUL	3	76	466	0	42	0	0	2	0
AUG	0	72	463	0	42	0	0	1	0
SEP	0	73	459	0	47	0	0	2	0
OCT	0	76	457	0	42	0	0	2	0
NOV	0	73	454	0	45	0	0	1	0
DEC	0	74	454	0	45	0	0	1	0

1986	INCR	TREQ	A101	A104	MINR	MAJR	UNSC	REFB	CONV
JAN	0	75	452	0	43	0	0	2	0
FEB	0	74	453	0	40	0	0	1	0
MAR	0	73	453	0	46	0	0	2	0
APR	0	74	454	0	42	0	0	2	0
MAY	0	75	455	0	44	0	0	1	0
JUN	0	72	457	0	42	0	0	2	0
JUL	0	72	461	0	38	0	0	1	0
AUG	0	65	466	0	44	0	0	1	0
SEP	1	77	467	0	49	0	0	2	0
OCT	3	80	469	0	29	2	0	2	0
NOV	0	71	481	0	34	9	0	1	0
DEC	0	71	486	0	29	12	0	2	0

1987	INCR	TREQ	A101	A104	MINR	MAJR	UNSC	REFB	CONV
JAN	0	78	495	0	28	12	0	1	0
FEB	0	80	504	0	24	18	0	1	0
MAR	9	89	508	0	29	25	0	3	0
APR	16	105	510	0	27	30	0	2	0
MAY	14	119	508	0	26	28	0	1	0
JUN	8	127	508	0	23	29	0	1	0
JUL	9	136	509	0	26	31	0	1	0
AUG	19	155	508	0	21	35	0	2	0
SEP	17	172	510	0	21	29	0	1	0
OCT	17	189	510	0	22	33	0	2	0
NOV	16	205	510	0	21	30	0	2	0
DEC	11	216	510	0	22	31	0	1	0

1988	INCR	TREQ	A101	A104	MINR	MAJR	UNSC	REFB	CONV
JAN	6	222	509	0	20	38	0	2	21
FEB	11	233	508	0	13	34	0	0	34
MAR	10	243	509	0	14	35	0	2	37
APR	11	254	518	22	17	33	0	2	35
MAY	16	270	525	32	17	35	0	1	36
JUN	10	280	536	45	17	34	0	2	36
JUL	16	296	546	61	17	37	0	2	56
AUG	11	307	554	75	15	27	0	1	24
SEP	30	337	572	90	14	27	0	0	14
OCT	13	350	588	100	18	35	0	2	18
NOV	24	374	598	117	11	24	0	2	11
DEC	6	380	606	142	19	35	0	2	19

1989	INCR	TREQ	A101	A104	MINR	MAJR	UNSC	REFB	CONV
JAN	0	377	602	165	15	24	0	2	15
FEB	0	375	610	189	14	25	0	1	14
MAR	0	371	609	208	20	33	0	1	10
APR	0	364	608	228	10	25	0	2	19
MAY	0	371	609	250	4	65	0	2	35
JUN	0	357	595	271	0	29	0	1	27
JUL	0	343	610	288	2	48	0	2	30
AUG	0	338	604	295	2	47	0	2	29
SEP	0	324	614	308	0	32	0	0	28
OCT	0	327	624	318	3	61	0	1	30
NOV	0	318	617	331	4	24	0	2	4
DEC	0	339	634	348	16	36	0	2	16

1990	INCR	TREQ	A101	A104	MINR	MAJR	UNSC	REFB	CONV
JAN	0	337	624	357	17	30	0	2	45
FEB	0	356	623	385	11	18	0	1	30
MAR	0	364	621	405	32	33	0	2	40
APR	0	359	620	408	12	16	0	0	12
MAY	3	383	636	423	31	20	0	1	52
JUN	0	382	626	426	26	23	0	1	42
JUL	15	398	628	447	18	13	0	2	33
AUG	9	407	638	455	39	26	0	1	60
SEP	0	399	635	455	16	11	0	2	29
OCT	0	400	651	472	33	18	0	1	52
NOV	0	390	647	469	26	18	0	1	45
DEC	0	339	652	483	21	12	0	2	35

1991	INCR	TREQ	A101	A104	MINR	MAJR	UNSC	REFB	CONV
JAN	0	327	658	493	37	25	0	2	64
FEB	0	290	652	496	16	9	0	2	27
MAR	0	256	663	513	29	20	0	1	50
APR	0	226	648	514	0	46	0	2	43
MAY	0	163	651	528	15	16	0	1	15
JUN	0	157	657	541	58	0	0	2	60
JUL	0	140	658	548	25	0	0	1	26
AUG	0	126	642	576	32	0	0	2	34
SEP	0	118	645	593	28	0	0	2	30
OCT	0	95	657	611	23	6	0	2	31
NOV	0	107	657	624	34	13	0	0	47
DEC	0	110	653	637	7	3	0	2	12

1992	INCR	TREQ	A101	A104	MINR	MAJR	UNSC	REFB	CONV
JAN	0	102	672	663	25	2	0	2	29
FEB	0	83	675	680	10	0	0	1	11
MAR	0	65	701	708	8	0	0	2	10
APR	0	72	711	728	26	0	0	2	28
MAY	0	64	720	749	8	0	0	1	9
JUN	0	63	748	779	24	0	0	1	25
JUL	0	75	742	793	32	0	0	2	34
AUG	0	87	742	812	29	0	0	1	30
SEP	0	111	702	826	10	0	0	2	12
OCT	0	89	721	854	3	0	0	2	5
NOV	0	62	750	884	18	0	0	1	19
DEC	0	54	702	901	19	0	0	2	21

1993	INCR	TREQ	A101	A104	MINR	MAJR	UNSC	REFB	CONV
JAN	0	60	644	925	1	0	0	1	2
FEB	0	58	675	955	2	0	0	1	3
MAR	0	40	656	979	22	0	0	2	5
APR	0	51	687	990	25	0	0	1	1
MAY	0	74	663	980	41	0	0	2	6
JUN	0	83	694	973	33	0	0	1	0
JUL	0	82	724	966	37	0	0	1	0
AUG	0	79	755	956	35	0	0	1	0
SEP	0	76	786	948	46	0	0	2	0
OCT	0	94	816	918	35	0	0	1	0
NOV	0	96	847	926	10	0	0	2	0
DEC	0	58	877	940	17	0	0	1	0

1994	INCR	TREQ	A101	A104	MINR	MAJR	UNSC	REFB	CONV
JAN	0	57	903	949	31	0	0	2	21
FEB	0	63	0	959	15	0	0	0	0
MAR	0	66	0	971	17	0	0	2	0
APR	0	66	0	985	11	0	0	1	0
MAY	0	43	0	1001	13	0	0	1	0
JUN	0	44	0	1021	18	0	0	1	0
JUL	0	53	0	1030	34	0	0	2	0
AUG	0	73	0	1026	26	0	0	0	0
SEP	0	72	0	1029	27	0	0	1	0
OCT	0	69	0	1028	28	0	0	1	0
NOV	0	75	0	1030	20	0	0	2	0
DEC	0	65	0	1029	36	0	0	1	0

1995	INCR	TREQ	A101	A104	MINR	MAJR	UNSC	REFB	CONV
JAN	0	77	0	1030	2	0	0	1	0
FEB	0	56	0	1057	10	0	0	0	0
MAR	0	49	0	1071	44	0	0	2	0
APR	0	76	0	1056	23	0	0	1	0
MAY	0	77	0	1051	42	0	0	1	0
JUN	0	77	0	1044	15	0	0	1	0
JUL	0	72	0	1060	39	0	0	2	0
AUG	0	94	0	1035	52	0	0	1	0
SEP	0	106	0	1023	22	0	0	0	0
OCT	0	81	0	1017	64	0	0	1	0
NOV	0	98	0	986	30	0	0	2	0
DEC	0	99	0	986	40	0	0	1	0

1996	INCR	TREQ	A101	A104	MINR	MAJR	UNSC	REFB	CONV
JAN	0	94	0	957	55	0	0	1	0
FEB	0	103	0	940	24	0	0	0	0
MAR	0	86	0	935	65	0	0	2	0
APR	0	104	0	898	34	0	0	0	0
MAY	0	105	0	897	40	0	0	1	0
JUN	0	92	0	870	54	0	0	1	0
JUL	0	104	0	859	3	0	0	1	0
AUG	0	75	0	870	62	0	0	1	0
SEP	0	99	0	838	29	0	0	1	0
OCT	0	97	0	842	30	0	0	0	0
NOV	0	68	0	835	32	0	0	2	0
DEC	0	74	0	832	27	0	0	1	0

1997	INCR	TREQ	A101	A104	MINR	MAJR	UNSC	REFB	CONV
JAN	0	72	0	829	49	0	0	1	0
FEB	0	90	0	809	16	0	0	0	0
MAR	0	78	0	829	12	0	0	1	0
APR	0	59	0	829	30	0	0	0	0
MAY	0	59	0	847	3	0	0	0	0
JUN	0	47	0	866	26	0	0	0	0
JUL	0	59	0	867	17	0	0	0	0
AUG	0	58	0	891	10	0	0	0	0
SEP	0	55	0	894	43	0	0	0	0
OCT	0	76	0	892	29	0	0	0	0
NOV	0	73	0	890	20	0	0	0	0
DEC	0	59	0	911	4	0	0	0	0

1998	INCR	TREQ	A101	A104	MINR	MAJR	UNSC	REFB	CONV
JAN	0	37	0	937	12	0	0	0	0
FEB	0	44	0	947	24	0	0	0	0
MAR	0	54	0	955	6	0	0	0	0
APR	0	47	0	983	2	16	0	0	0
MAY	0	42	0	990	7	18	0	0	0
JUN	0	76	0	990	1	36	0	0	0
JUL	0	93	0	984	8	37	0	0	0
AUG	0	114	0	969	0	35	0	0	0
SEP	0	112	0	960	2	32	0	0	0
OCT	0	106	0	952	1	41	0	0	0
NOV	0	121	0	933	2	38	0	0	0
DEC	0	127	0	921	1	19	0	0	0

1999	INCR	TREQ	A101	A104	MINR	MAJR	UNSC	REFB	CONV
JAN	0	109	0	939	2	17	0	0	0
FEB	0	81	0	950	0	10	0	0	0
MAR	0	59	0	967	19	12	0	0	0
APR	0	75	0	957	5	18	0	0	0
MAY	0	74	0	969	2	13	0	0	0
JUN	0	60	0	984	1	16	0	0	0
JUL	0	57	0	1002	1	11	0	0	0
AUG	0	53	0	1013	1	41	0	0	0
SEP	0	78	0	1007	1	17	0	0	0
OCT	0	89	0	1008	1	29	0	0	0
NOV	0	94	0	1010	1	35	0	0	0
DEC	0	98	0	1009	1	12	0	0	0

2000	INCR	TREQ	A101	A104	MINR	MAJR	UNSC	REFB	CONV
JAN	0	85	0	1017	2	44	0	0	0
FEB	0	102	0	1007	1	0	0	0	0
MAR	0	75	0	1033	0	8	0	0	0
APR	0	62	0	1051	1	40	0	0	0
MAY	0	78	0	1034	2	25	0	0	0
JUN	0	103	0	1041	1	39	0	0	0
JUL	0	113	0	1025	1	21	0	0	0
AUG	0	95	0	1039	1	30	0	0	0
SEP	0	97	0	1023	1	59	0	0	0
OCT	0	114	0	1003	1	13	0	0	0
NOV	0	120	0	1013	1	54	0	0	0
DEC	0	118	0	975	1	34	0	0	0

PART 3

0 MISSILES WERE BROKEN AT DEPLOYMENT BASES THROUGHOUT THE LIFE OF THE SYSTEM. THEY AVERAGED 0. DAYS ON BASE (BRO-KEN) BEFORE BEING REPLACED FOR A TOTAL OF 0. DAYS IN INOPERABLE STATUS.

6936 MISSILES WERE KEPT IN A DEPLOYED STATUS, UP TO OR PAST THEIR WARRANTED LIFE, THROUGHOUT THE LIFE OF THE SYSTEM. THEY AVERAGED 36.62 DAYS ON BASE (OVERDUE) BEFORE BEING REPLACED FOR A TOTAL OF 253999.00 DAYS OVERDUE.

Appendix B: Glossary of Selected SIMSCRIPT Terms

Refer to APPENDIX E: Program Listing for the context in which the following terms are used. Words appearing in all CAPITALS are terms used in the program.

ATTRIBUTES: ATTRIBUTES are the "memory cells (19:3)" of ENTITIES and PROCESSES. They define the characteristics of each ENTITY or PROCESS. Attributes may be set to different values or examined by the actions of the program.

ENTITIES: ENTITIES are the model's objects. They represent the devices or objects that exist in the real system. Like a subscripted variable, each ENTITY can represent many values at the same time, by the carrying of ATTRIBUTES (18:222).

FOR EACH ... OF (SET): This phrase causes a group of statements to be executed for each object stored in the specified SET. It can be modified with WHILE, UNTIL, WITH OR UNLESS phrases so that the statements are only executed for certain of the objects, or UNTIL a specific object is encountered (19:151).

PROCESSES: PROCESSES are the sequence of actions that an object undergoes as it passes through the model. There may be many copies of each PROCESS in existence at any one time, each processing a different object. "The process routine may test for system conditions and take alternative courses of action (15:2-3)".

RESOURCES: RESOURCES are "used" by the model's objects. If the number of units of the RESOURCE requested by an object are available, the units are taken and held by the object until it releases them. If the RESOURCEs are not available, the requesting object is placed in a queue to wait for the units to become available (15:2-4).

ROUTINES: ROUTINES are similar to subroutines in that values may be passed to them, the ROUTINE may perform some operation, and values may be returned by them to the CALLing ROUTINE or PROCESS.

SCHEDULE/ACTIVATE: SCHEDULE and ACTIVATE both "activate the future occurrence of a process (19:59)" by assigning a future time (the AT, IN or NOW phrase) for the PROCESS to start. At that moment in (SYSTEM) time, the previously scheduled PROCESS begins to occur and effect the system.

PROCESSes may be suspended for a certain amount of time after they are started by using the WAIT or WORK statements.

SETS: SETS are ordered groups of objects that exist in the model. ENTITIES "can either own or belong to a set (15:3-35)." ATTRIBUTES that order the SET are automatically created by the SYSTEM. These "owner" ATTRIBUTES describe the number of objects currently in the SET, and the first and last member of the SET. Member ATTRIBUTES describe the predecessor and successor objects of each SET member, and identify if the object is currently a member of that SET. Objects can be placed in or removed from SETS by the actions of the program (FILE and REMOVE phrases), and the order of that placement can also be changed by program actions(15:3-35-38).

SYSTEM: The SYSTEM is a term used for the operating system. It is in existence to act as the OWNing agency for various SETS that are not OWNed by other ENTITIES. It may also have its own ATTRIBUTES (19:281-286).

TALLY/ACCUMULATE: The TALLY statement is used to collect various types of statistics on system VARIABLES These statistics include the SUM, NUMBER, MEAN, VARIANCE and many other values. The ACCUMULATE statement calculates similar statistics, but weights the resultant value with the amount of time the variable remains in any given state. Thus the TALLY of 2 for 1 minute, and 6 for 2 minutes would be 4, while the ACCUMULATE for these values would be 5.

VARIABLES:

GLOBAL: Global VARIABLES are names of memory locations that are known (can be examined and/or changed) throughout the entire program. VARIABLES are made global by declaring them in the PREAMBLE section (19:44).

LOCAL: (Recursive) local VARIABLES are names of memory locations that are known only within the particular copy of the PROCESS they appear in. (Saved) local VARIABLES are known throughout every copy of the particular PROCESS they are used in. Local VARIABLES are declared in the PROCESS itself, and override the global VARIABLE of the same name.

Appendix C: <u>Model Operation and Decision Logic</u>

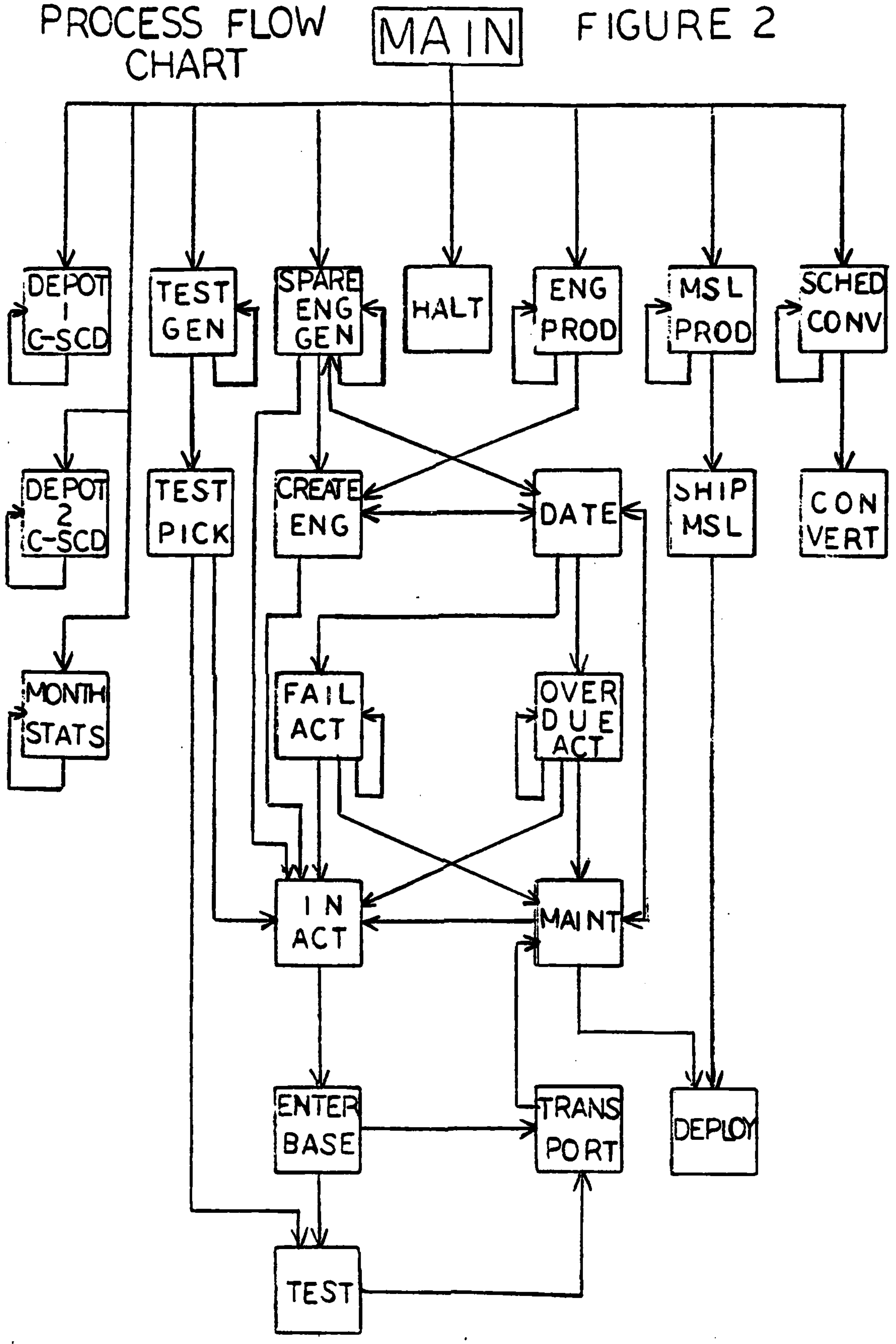
This appendix explains the operation and decision logic of the SIMSCRIPT model listed in Appendix E: Program Listing.

The purpose of this appendix is to explain the computer model to those users who desire a detailed understanding of the model's operation and decision logic. This depth of understanding is necessary if the user desires to modify the model to reflect changing parameters or system structure, to gather information on aspects of the system not explicitly addressed in this model, or to experiment with those factors of the model in being, but not the subject of this research.

Those statements in the model which are self-explanatory as to their underlying assumptions, their purpose and their operation are not addressed in this appendix. To help guide the user through the sequential flow of the model, Figure 2: Process Flow Chart should be examined. This figure depicts the possible "paths" through the model. It shows all the processes that may <u>activate</u> a given process and all the processes that may be <u>activated by</u> that same process. In addition, the interested user should refer to Appendix B: Glossary of SIMSCRIPT Terms for an explanation of some of the terms peculiar to SIMSCRIPT, and to Appendix F: Explanation of Program Terms, for a description of the variable names, sets, entities and other items used in the program. For a more detailed explanation of the

PROCESS FLOW CHART

FIGURE 2



SIMSCRIPT language, refer to the three texts by C.A.C.I. (references 15, 18 and 19).

The line numbers cited on the left hand side of this appendix refer to those in Appendix E. The topic headings refer to the main sections and processes of the program in Appendix E.

PREAMBLE

94	See routine DATE line 469.
95	See process IN.ACTION lines 629-630
(144-155)	These statements set up the system routines needed to automatically gather statistics on the number of spare, broken, and overdue engines.
144-147	MAX.PER.MONTH is the maximum number of spares needed during a one-month period, after which it is reset to zero and recalculated for the next month (see MONTHLY.STATISTICS). LIFE.MAX is the maximum number of spares needed over the <u>life</u> of the system. (see note 1 and TALLY/ACCUMULATE in Appendix B)
148-151	BROKENUM is the number of engines broken over the life of the system. BROKESUM is the summation of the time all engines spend on base while broken. (see TALLY/ACCUMULATE in Appendix B)
148-151	OVERNUM is the number of engines overdue over the life of the system. OVERSUM is the summation of the time all engines spend on base while overdue (see TALLY/ACCUMULATE in Appendix B)

MAIN

166	SIMU7 is the file containing the variable input parameters. SIMU8 is the file to which all program output is written.

192-193 The maximum number of engines that can be serviced at any one time by each DEPOT is set at this point. This capacity can be changed at a later date by the DEPOT(1/2).CAPACITY.SCHEDULE process. The original capacity of 200 units is equivalent to an "unlimited" service capacity, since the maximum number of engines the depot would be required to service at any one time is 115 spares and 4 to 5 OTL test engines.

MAINTENANCE

216-217 This months (COUNT = month) MAINTENANCE.RECORD is updated by tallying the type of maintenance to be performed (INDEX) on the incoming engine. INDEX is obtained from the TYPE.SERVICE attribute of the engine.

218-219 After this point, the type of service indicator (INDEX) is set to indicate LIMITED (1) or FULL (2) for use in controlling which engines are converted. Since all types of maintenance except limited are of the longer duration, they are treated as fulls.

221-222 These statements insure only the F107-101 type engine is converted, the current simulation time (TIME.V) is past the start-conversion date, and the conversion quota for that engine type (MAY.CONVERT) has not been exhausted. The conversion period is open-ended on the back side to insure all engines will eventually be converted, since some of them may be out of cycle during the scheduled conversion period.

224 All conversion are completed at DEPOT1 only.

229 The time required to REPAIR and engine is equal to the time required for a FULL overhaul.

231,235 After an overhaul of one type (REPAIR/FULL, or LIMITED), the next overhaul will be of the opposite type (LIMITED or FULL).

238 Maintenance performed on engines that are converted is double-counted. Both the scheduled maintenance (FULL, LIMITED, REFURB, or REPAIR) and the actual maintenance (CONVERSION) is recorded.

241,244 Both CONVERSIONs and REFURBishments require a subsequent LIMITED overhaul.

249 The engine's attributes indicating when maintenance was performed and when it will be due again are updated in routine DATE.

250 If the engine has been counted as a required spare, this statement removes it from the spares required counter and updates its attributes to reflect that it is no longer a required spare. (see note 1)

251-257 If the engine came from the PRODUCTION.POOL, it had failed or became overdue before its assembly with an airframe and before being shipped to a base. These statements return it to the PRODUCTION.POOL to eventually by shipped out to a base.

259-263 If the engine came from an OTL missile, it must be replaced in that missile and returned to its originating base. It is not a spare engine available for general use. (see note 2)

267-277 If the repaired engine is the only available spare, indicated by both DEPOT's repaired stocks (N.REPAIRED.SET) being empty (0), and a demand already exists for a spare engine (there is an engine in the TAKE.SET or EARLY.OVERHAULs are authorized), the engine is immediately sent to be used as a spare (IN.ACTION is activated with a specific MSL). Otherwise, the engine is placed in the appropriate repaired stock (REPAIRED.SET) and the system is notified of its availability (IN.ACTION is activated without a specific MSL).

<u>SPARE.ENGINE.GENERATION</u>

284-286 This process is active for a two-year period (1982 and 1983), twelve months a year (1 to 12) and five times per month (1 to 5).

288 If the number of spare engines to be produced (NUMBER.SPARES) has been reached (L), the process is terminated.

291 This statement simulated the random nature of the spare engine production schedule, "creating" the engine within plus-or-minus two days (uniformly distributed) of its scheduled production date.

293-298	These statements initialize the attributes of the newly created engine, defining its characteristics.
299	All spare engines are produced at the Williams plant (also DEPOT1).
300-310	See MAINTENANCE, lines 267-277.

TEST.GENERATION

320	Reads from input file SIMU11.
320-326	Reads the number of days past simulation start-up to schedule a test (TEST.DAYS) and the type of test to be scheduled (TEST.TYPE). Then it schedules the process TEST.PICK to select a test engine/missile. This is repeated for the number of tests to be performed (NUMBER.TESTS).

TEST.PICK

332-333	Selects a missile (PICK) from the set of operational missiles (ALERT.MISSILE.SET) for testing. The missile selected is the oldest one that has not already been selected for testing (TEST.STATUS = NONE) and is not already being processed for an overhaul or repair (CLAIM not equal to TAKEN).
334	Assigns the type of test the engine will undergo to TEST.STATUS.
(335-345)	For EVIP tests only.
336-344	See MAINTENANCE lines 267-277, but substitute TAKE.SET for REPAIRED.SET. (see note 1)
(346-360)	For OTL and JTA tests only.
346-347	Missiles selected for testing are not counted as overdue (they are not part of an operational force).
350	OTL and JTA missiles are automatically removed from the ALERT.MISSILE.SET since the do not require a spare engine.

354-356	OTL missiles are shipped to the test site for testing. (see note 2)
357-359	JTA missiles are considered destroyed at this point since they no longer add to or subtract from the variables of interest in the simulation

MISSILE.PRODUCTION

368	Reads from input file SIMU13.
376	If no missiles are to be shipped that month, skip to process the next month.
377	Break the shipment of missiles for the month (MSL.PRODUCTION.NUMBER) into four equal shipments (NUMBER.TO.SEND).
378-382	For the first three shipments of the month (days 7,14, and 21, plus or minus 2 days) indicate the NUMBER.TO.SEND to process SHIP.MISSILE.
383	Ship the remainder of missiles for the month on the twenty-eighth day of the month.

SHIP.MISSILE

393	Ship the number of missile received from MISSILE.PRODUCTION, one at a time.
396-398	If a missile is available from the missile stockpile (PRODUCTION.POOL), select it.
399-400	If no missiles are available, wait one day and try it again.
405-406	If the current base's quota of missiles (BASE.MISSILE.RQMT) has not been filled (equal to BASE.MISSILE.COUNTER), ship a missile (activate a DEPLOY) to that base and increase that base's total by one. (see note 2)
407-410	If the currents base's quota has been filled, begin shipping to the next base.

ENGINE.PRODUCTION

420	Reads from input file SIMU15.
428	If no engines are to be produced that month, skip to the next month.
429	Calculate the time interval (DATE.TO.MAKE.ENGINE) between each engine's production for the month.
430-434	Schedule a CREATE.ENGINE every time interval until the number of engines to be produced that month has been reached.

CREATE.ENGINE

442-448	See SPARE.ENGINE.GENERATION lines 293-298.
449-450	Original-production engines are shipped to BOEING and place in its missile PRODUCTION.POOL. Since engine production is so far ahead of missile production, and is projected to stay so, no time delay is required for this shipment.

DATE

457	The current simulation time (TIME.V) is recorded as the production/overhaul date of the engine in the attribute START.DATE.
458	If a randomly selected number between zero and one is less than the ENGINE.FAILURE.RATE, the engine will be scheduled to fail prematurely. This process simulates the probability of a random engine failure, one of the variable factors in the model.
459	The length of time an engine lives before a failure is assumed to be uniformly distributed over the normal life for that type of engine (TERM). The failure date is calculated as the current time (TIME.V) plus a uniform portion of its normal life [RANDOM.F(STREAM3) * TERM(TYPE(ENGINE)-100)].
461	Since the engine will fail, its next service type will be REPAIR.

462	The process FAILURE.ACTION is activated on the date the engine will fail. This is the "flag" that lets the system know an engine has failed. This is based on the assumption that the engine's failure is detected on that date. Thus the model keys on the failure detection date, not the actual (unknown) failure date. The failure rates used by the model are actually failure detection rates.
464	If the random number selected is greater than or equal to the FAILURE.RATE, the missile will last its normal life.
466	For an engine that lasts its normal life, the process OVERDUE.ACTION is activated TRANSPORT.DAYS before the engine's overdue date. This parallels the real-world system, where a spare engine would be ordered for a soon-to-be-overdue engine early enough to arrive on or before the date it is needed.
469	The RANK.DATE is set to the normal life of both failure and overdue engines. Engines are then placed in the ALERT.MISSILE.SET oldest RANK.DATE first. This sequences all missiles for test selection from the ALERT.MISSILE.SET based on their normal lifetime. Since a failure is not detected until it happens, test selection should not be biased because of an event that has not yet happened (the future failure). Test selection is based on using the engine closest to its expiration date (the oldest engine of its type on base). If engines were sequenced according to their DATE.EXPIRES, only soon to fail engines (if any were on base) would be selected for testing, obviously biasing the tests.

SCHEDULE.CONVERSION

477	Reads data from input file SIMU17.
479-482	Schedules a CONVERT the first day of every month from January 1988 through December 1991.

CONVERT

488	Reads data from input file SIMU17.

489-491	Reads each month's quota for conversions for both types of service (LIMITED = LIMITED quota; FULL, REPAIR, REFURBish = FULL quota), and adds these quotas to the amount remaining from last month. (see MAINTENANCE lines 218-222)

TEST

515	Places the engine in the TEST.SET prior to testing. This identifies the engine as being tested if it fails or becomes overdue while undergoing a test.
516-520	Simulates testing by delaying processing for the amount of time required for the test (EVIP.TEST or OTL.TEST days)
521	Checks to see if the engine is still in the TEST.SET. If not, the engine has failed during the test (FAILURE.ACTION has occurred), and the engine has been previously removed from testing. If so, the process is terminated.
522-526	If this is an OTL missile that has been recovered after testing (RANDOM.F(STREAM1) is greater than the RECOVERY.FAILURE.RATE), or is an EVIP test engine, ship the engine/missile to the depot for service.
527-529	Missile is an OTL that was not recovered and is thus destroyed.

OVERDUE.ACTION

535-537	If the DATE.EXPIRES of the engine being processed is not equal to the current simulation time (TIME.V), this OVERDUE.ACTION is the originally scheduled process for an engine that has been selected for testing, and refurbished (thus receiving a new DATE.EXPIRES) before its original overdue date. These statements terminate this "ghost" OVERDUE.ACTION.
538-540	If the CLAIM of the engine is TAKEN, the engine is being worked on by another process (possibly TEST.PICK). This OVERDUE.ACTION must be delayed until the previous one is complete (a maximum of

TRANSPORT.DAYS). This precludes having two processes trying to handle an engine at the same time (which causes the system to make a "double" of the engine).

(541-553)	For those engines in the ALERT.MISSILE.SET when they become overdue:
545-552	See MAINTENANCE lines 267-277, but replace REPAIRED.SET with TAKE.SET.
554-557	For those engines in the depot stock of repaired engines when they become overdue: remove them from the repaired stock and initiate servicing at DEPOT1.
558-562	For those engines at BOEING when they become overdue: remove them from the BOEING stock and initiate servicing at DEPOT1.
563-567	If an engine is INTRANSIT (being shipped from one place to another) when it becomes overdue: it cannot be handled by the model until it arrives at it destination (unknown by the model). Delay this OVERDUE.ACTION until the engine arrives at its destination (its arrival time is a maximum of TRANSPORT.DAYS away). Then reinitiate this process from the beginning.

<u>FAILURE.ACTION</u>

576-578	See OVERDUE.ACTION lines 535-537.
529-581	See OVERDUE.ACTION lines 538-540.
(582-595)	For those engines that have failed while in the ALERT.MISSILE.SET:
586	As opposed to overdue engines, failed engines cannot be used as operational missiles and are thus removed from the ALERT.MISSILE.SET.
587-594	See OVERDUE.ACTION lines 545-547 and 548-552.
596-600	See OVERDUE.ACTION lines 554-557.
601-605	See OVERDUE.ACTION lines 558-562.

606-610	As opposed to an engine that becomes overdue while being tested (where the overdue is ignored), an engine failure occurring in a test must be removed from the test and shipped to DEPOT1 for servicing. This assumes that and engine will not continue to be tested after it has failed. (see note 2)
611-615	See OVERDUE.ACTION lines 563-567.

<u>IN.ACTION</u>

626	If an outgoing engine (one due for replacement) has already been identified by the system (FLAG is greater than 100), there is no need to search for another engine to replace. Skip the following statements and transfer control to line 646.
626-630	If an engine <u>has</u> <u>not</u> been identified for replace the system must initiate a search for the most eligible engine. Since engines are placed into the TAKE.SET with a priority code attribute (STATUS), the most eligible engine will be at the beginning of the TAKE.SET. Lines 629 and 630 look in the beginning of the TAKE.SET and find the first engine that is not TAKEN and assigns its memory location to the variable FLAG.
631-633	If such an engine has been found, the outgoing engine has been identified and control passes to line 646.
634	If there is no eligible engine in the TAKE.SET and EARLY.OVERHAULs are authorized:
635-638	A search is made of the ALERT.MISSILE.SET to find the engine closest to its RANK.DATE, not CLAIMed, and with no STATUS code (indicating an engine that has been selected for a test, is overdue, or broken). If such an engine has been found, the outgoing engine has been identified and control is passed to line 646.
640-645	If no eligible engine has been identified for replacement, the incoming spare engine, if any (indicated by MSL not equal to zero), is shipped back to DEPOT1 as a spare resource and the process is terminated.

646-649 Once the outgoing engine has been identified, a check is made (MSL greater than 100) to see in an incoming spare engine has also been identified. If so, control is passed to line 663.

650-662 If no incoming spare engine has been identified, a search of the repaired stocks is made to locate one. Priority is given for selection from DEPOT1. If its stock if spare engines is equal to or greater then DEPOT2's (and not zero), the oldest spare engine is selected from the DEPOT1 repaired engine stock. If DEPOT2 has more spares than DEPOT1 (but not zero), the oldest spare engine is selected from the DEPOT2 repaired engine stock. If no spare is available from either stock, the outgoing engine is placed in the TAKE.SET (if it is not already in it and it is not an EARLY.OVERHAUL engine [STATUS = NONE]), and the process is terminated.

663-667 If both an outgoing engine and an incoming engine have been identified, the outgoing engine's CLAIM is set to TAKEN (to preclude its selection for removal by another process) and an ENTER.BASE is scheduled in TRANSPORT.DAYS. (see note 2)

<u>ENTER.BASE</u>

674-678 Even though an engine has been identified for replacement by the process IN.ACTION, it has been TRANSPORT.DAYS since that has happened. During this time, some other engine at the same base with a higher priority could have become eligible for replacement. If so, and no spare engine was immediately available for its replacement, it would have been placed in the TAKE.SET. So a check must be made of the TAKE.SET. If it is empty, there can be no other engines of a higher priority, so the previously identified engine (FLAG) will be replaced (ENG.OUT = FLAG) and control is passed to line 689.

679-682 If other engines are in the TAKE.SET, it must be checked for higher priority engines. This is accomplished by placing it in the TAKE.SET (if it is not already there and it is not an EARLY.OVERHAUL engine), to assume its rightful place in the "replacement pecking order". Its CLAIM must first be cleared to allow it to be chosen from the TAKE.SET.

683-687	A check is then made of the TAKE.SET to select the highest priority engine (at the previously identified engine's base) for replacement. If one is found (which could be the previously identified one), its memory location to the variable ENGINE.OUT.
(689-710)	These statements process the outgoing engine.
690-693	An outgoing engine identified for an EVIP test (TEST.STATUS = EVIP), that has not failed, is shipped to the test site. (see note 2)
694-695	An engine selected for an EVIP test that has failed (STATUS = BROKE) is sent to DEPOT1 for servicing instead of the test site.
696-702	The counter for the number of engines overdue or broken (as appropriate) is decremented by one. The engine is removed from the base and the counters only track those on a base.
704-709	The outgoing engine is removed from various on-base set if it is in them.
711-713	The incoming spare engine's location is set to the base of the engine it is replacing, and the engine (now in an operational missile) is placed in the ALERT.MISSILE.SET.

<u>DEPOT1.CAPACITY.SCHEDULE</u>

723	Reads data from input file SIMU19.
724	Continues on only if there is more data to read.
725	Reads in the new servicing capacity for DEPOT1 (CAPACITY) and the date of the change in CAPACITY.
726	Delays until change date.
728	Releases the number of units of DEPOT1 capacity it previously preempted (OLD). This gives the maximum capacity back to the DEPOT.
729	Preempts CAPACITY units of DEPOT1 servicing capacity by requesting them with a high (#1) priority. For example, if the new capacity was supposed to be 50 units, it would preempt 200-50 or 150 units, leaving 50 units for the DEPOT to use. If 150

units were not available at this time, it would wait and preempt them as they became available.

730 The new CAPACITY is assigned to the variable OLD to be released in the next change.

733 Control is passed back to line 741 to check for further changes.

DEPOT2.CAPACITY.SCHEDULE

738-753 See DEPOT1.CAPACITY.SCHEDULE lines 719-734

HALT A sample of the output generated by this process is presented in Appendix A: Sample Output Summary. The output has been reformatted for inclusion in this paper. The content, however, has not changed from that generated by this process.

839-851 The counters BROKENUM and OVERNUM are the number of engines broken and overdue during the life of the system. BROKESUM and OVERSUM are the total amount of time broken and overdue accumulated by these engines. The system generated routines used track these values double-counts the engines in question. One is counted when an engine is added to the counter and one is counted when the same engine is subtracted from the counter. Thus the actual number of engines broken or overdue is one-half of that indicated by the counter. Variables B2 and O2 contain the correct numbers. BROKESUM/B2 and OVERSUM/O2 give the average time broken or overdue.

852 After all the statistics are printed out the program is terminated.

MONTHLY.STATISTICS

860 The month being processed (the variable COUNT) is incremented each time the process is activated (each month).

861-862 The local variables are reset for each months calculations.

863-876 The average age of each type of engine in the ALERT.MISSILE.SET is calculated each month. Each engine in the ALERT.MISSILE.SET is examined and its age (current time - START.DATE) is summed in the variables TOTAL1 (for 101's) or TOTAL4 (for 104's). The number of each type of engine is tabulated in the variables DIVISOR1 and DIVISOR4. The summed ages of each type are then divided by the total number of each type, giving average ages. These values are then assigned to the AVERAGE.AGE array for that month. To preclude division by zero (if there are no engines of a given type), any zero-valued divisor is set equal to 1.

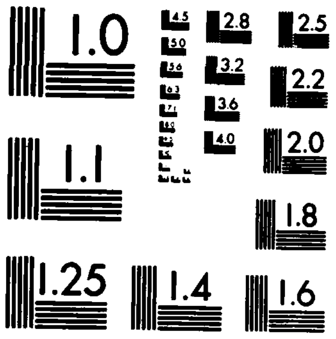
877 The maximum number of engine required by the system over the last month (see note 1) is assigned to the SPARES array for that month. Since this process is activated on the first day of each month, the statistics given for that month are derived from and apply to the previous month's data. The statistics in the model, as those in the real world, are always a month behind.

878-881 If the maximum number of engine required for this month is greater than any previous month, the increase in engines required is recorded in the SPARES array for the month, and this month's spares requirement is saved in the variable LAST1 to be used as the new comparison value in the upcoming months.

882 All monthly statistics are reset to zero.

SHIP

887-888 The time the engine will arrive at its destination (the current simulation time [time.v] plus transportation time [TRANSPORT.DAYS]) is assigned to the engine's ARRIVAL attribute, and the engine's SHIPPING.STATUS is changed to reflect its shipment (INTRANSIT). (see note 2)


1.0
1.1
1.25
1.4
1.6
1.8
2.0
2.2
2.5
2.8
3.2
3.6
4.0

DEPLOY

895	Deployment base numbers range from 3 to 8, but the SHIP.MISSILE process ranges them from 1 to 6, this line adjusts the difference.
896-897	The incoming missile arrives and is placed in the ALERT.MISSILE.SET as an operational asset.

ADD.TO.SPARES.COUNT

903-906	If the engine is not already counted as a spare or as requiring a spare, it is added to the spares requirement counter (NUM.SPARE), and its SHIPPING.STATUS is changed to indicate it is a SPARE. (see note 1)

SUBTRACT.FROM.SPARE.STATUS

912-915	If the engine was previously counted as a spare (SPARE.STATUS = SPARE), its SPARE.STATUS is changed to indicate it is not a spare, and the spares counter (NUM.SPARES) is decremented by one.

REMOVE.FROM.REPAIRED.SET

921-935	This routine is called when an engine is in one of the DEPOT's REPAIRED.SETs but the particular set is not known. This is an idiosyncrasy of the SIMSCRIPT language where a entity is placed in a subscripted set (one of the sets owned by the DEPOTs), it cannot be directly examined to determine <u>which</u> set it is in. Without this knowledge, it cannot be successfully removed from the set. This routine overcomes this system fault by examining each set until it locates the engine in question. It then can be removed since it is now known which set it is in.

NOTE 1

The model was designed to indicate, at any given time, the number of engines that are being used as spares engines or require a spare engine. If an engine is being transported from a base to the DEPOT for repair, it is not available for use as an operational asset and thus is being used as a spare engine. The same is true for engines selected for EVIP tests, they are unavailable for operational use while being transported to and from the test site and while in the test program. They also require a spare engine to replace them in the missile they were removed from. Engines undergoing maintenance and servicing are also being used as spares.

If a demand for a spare engine exists, that cannot be immediately filled (as when an engine on base fails and there are no spares on that base), a demand for a spare engine is created beyond the number of spare engines being used as spares. This demand still exists while the replacement engine is being shipped to the base, but has not yet arrived.

Engines not counted as spares include those being transported to meet a demand (to do so would double-count the spares requirement, once for the engine being shipped and once for the demanding engine). Also excluded are OTL test engines. Since the airframe goes with the engine for an OTL test, there is no demand for a spare, nor a place for a spare to reside (as with the EVIP test engine). Finally those engines in missiles not yet deployed to bases are not being used as spares, nor creating a demand for a spare.

<u>NOTE 2</u>

To simulate the shipment of an engine or missile from one place to another, the PROCESS that controls the receiving activity for the engine is scheduled TRANSPORT.DAYS (the number of days it takes to ship an engine from one place to another) later. This simulates the time delay due to transportation. While the engine is being "transported", it is unavailable for processing by any other activity. To indicate this, its SHIPPING.STATUS is set to INTRANSIT, and the date it will arrive at its next activity is assigned to its ARRIVAL.TIME attribute. Both of these actions are accomplished by the SHIP routine.

Appendix D: Input Files SIMU7 and SIMU9

SIMU7

The following file is the first combination of the thirty-two different SIMU7 files. These files are used to change the independent variables in the simulation. The values on the left side of the file are changed to reflect the specifications shown in TABLE VII.

```
S01
4     DAYS.REQUIRED.TO.TRANSPORT.UNIT
40    DAYS.FOR.OTL.TESTING
40    DAYS.FOR.EVIP.TESTING
0.1   LOSS.RATE.FOR.OTL.TESTING
30    #.OF.DAYS.NEEDED.FOR.TEST.REFURBISHMENT
30    #.OF.DAYS.NEEDED.FOR.FULL.OVERHAUL
6     #.OF.DAYS.NEEDED.FOR.LIMITED.OVERHAUL
30    #.OF.DAYS.NEEDED.FOR.101.TO.104.CONVERSION
0.0   MISSILE.FAILURE.RATE
```

SIMU9

This file reflects values for factors that will be considered constants for this research effort. Future research efforts may change these values based on the assumptions made in their research.

```
1    MONTH.SYSTEM.STARTS
1    DAY.SYSTEM.STARTS
1981 YEAR.SYSTEM.STARTS
31   DAY.CONVERSIONS.END
1    MONTH.CONVERSIONS.END
1991 YEAR.CONVERSIONS.END
1    MONTH.SYSTEM.ENDS
1    DAY.SYSTEM.ENDS
2001 YEAR.SYSTEM.ENDS
1    MONTH.OCALC.OPENS.FOR.BUSINESS
1    DAY.OCALC.OPENS.FOR.BUSINESS
1989 YEAR.OCALC.OPENS.FOR.BUSINESS
1    MONTH.ENGINES.BEGIN.TO.BE.CONVERTED.TO.104'S
1    DAY.ENGINES.BEGIN.TO.BE.CONVERTED.TO.104'S
1988 YEAR.ENGINES.BEGIN.TO.BE.CONVERTED.TO.104'S
1    STREAM.NUMBER.1
2    STREAM.NUMBER.2
3    STREAM.NUMBER.3
4    STREAM.NUMBER.4
5    STREAM.NUMBER.5
912  NUMBER.OF.DAYS.WARRANTED.LIFE.FOR.A.101.ENGINE
1825 NUMBER.OF.DAYS.WARRANTED.LIFE.FOR.A.104.ENGINE
6    NUMBER.OF.BASES
115  NUMBER.OF.SPARES.ALLOWED.IN.SYSTEM
286  BASE.1.MISSILE.REQUIREMENTS
286  BASE.2.MISSILE.REQUIREMENTS
286  BASE.3.MISSILE.REQUIREMENTS
286  BASE.4.MISSILE.REQUIREMENTS
286  BASE.5.MISSILE.REQUIREMENTS
285  BASE.6.MISSILE.REQUIREMENTS
200  MAXIMUM.MONTHLY.CAPACITY.FOR.DEPOT1
200  MAXIMUM.MONTHLY.CAPACITY.FOR.DEPOT2
0    EARLY.OVERHAUL.OK?(YES=10,NO=0)
```

Appendix E: Program Listing

```
''ALCM ENGINE SIMULATION PROGRAM
''(BY D. RICKARD & T. SCHOMMER)
PREAMBLE
NORMALLY MODE IS INTEGER

''REDEFINE VARIABLES TO INSURE UNIQUE VALUES FOR SIMILAR
''VARIABLE NAMES
DEFINE SPARES                      TO MEAN S2PARES
DEFINE SPARES.COUNT                TO MEAN CSPARES
DEFINE ENGINE.PRODUCTION           TO MEAN PENGINE
DEFINE START.YR                    TO MEAN S1TART.YR
DEFINE START.MO                    TO MEAN S2TART.MO
DEFINE START.DATE                  TO MEAN D.START.DATE
DEFINE STREAM4                     TO MEAN 4STREAM4
DEFINE STREAM5                     TO MEAN 5STREAM5
DEFINE CONVERT.YR                  TO MEAN S1WITCH.YR
DEFINE CONVERT.MO                  TO MEAN S2WITCH.MO
DEFINE CONVERT.DAY                 TO MEAN S3WITCH.DAY
DEFINE STREAM1                     TO MEAN 1STREAM1
DEFINE STREAM2                     TO MEAN 2STREAM2
DEFINE TEST.PICK                   TO MEAN 4TEST.PICK
DEFINE DATE.EXPIRES                TO MEAN SDATE.EXPIRES
DEFINE ENGINE.ID.NUMBER            TO MEAN E.ID.NUMBER
DEFINE MISSILE.PRODUCTION          TO MEAN MISL.PROD
DEFINE BASE.MISSILE.COUNTER        TO MEAN C2BASE.MSL
DEFINE REPAIR                      TO MEAN R1EPAIRED
DEFINE NUMBER                      TO MEAN NUM
DEFINE ENGINE.IN                   TO MEAN ENG.IN
DEFINE ENGINE.OUT                  TO MEAN ENG.OUT
DEFINE CONVERSION                  TO MEAN CONV
DEFINE NUMBER.BASES                TO MEAN NUM.BASE
DEFINE TRANSPORT.DAYS              TO MEAN TRANS.DAYS
DEFINE MAINTENANCE.RECORD          TO MEAN MAIN.REC
DEFINE WORK.STATIONS.IN.USE        TO MEAN IWSTATION
DEFINE SPARE.STATUS                TO MEAN SPSTAT
DEFINE CONVERT                     TO MEAN CNVRT
DEFINE ENGINE.SEQUENCE.NUMBER      TO MEAN E.SEQ.NUM
DEFINE DEPOT1.CAPACITY.SCHEDULE TO MEAN D1SCHED
''***************************************************
PROCESSES
            INCLUDE     DEPOT1.CAPACITY.SCHEDULE,
                        DEPOT2.CAPACITY.SCHEDULE,
                        SPARE.ENGINE.GENERATION
              EVERY     ENTER.BASE     HAS AN A1, AN A2
              EVERY     MAINTENANCE    HAS A  B1, A B2
              EVERY     SHIP.MISSILE   HAS A  C1
              EVERY     TEST           HAS A  D1
```

```
'' PREAMBLE CONTINUED
PROCESSES
            INCLUDE    CONVERT,
                       CREATE.ENGINE,
                       ENGINE.PRODUCTION,
                       HALT,
                       MONTHLY.STATISTICS,
                       MISSILE.PRODUCTION,
                       SCHEDULE.CONVERSION,
                       TEST.GENERATION
              EVERY    DEPLOY           HAS A T1, A T2
              EVERY    IN.ACTION        HAS A Q1, A Q2
              EVERY    FAILURE.ACTION   HAS A R1
              EVERY    OVERDUE.ACTION   HAS A S1
              EVERY    TEST.PICK        HAS A F0
              EVERY    TRANSPORT        HAS A G1,A G2,
                                            A G3
''**************************************************
PERMANENT ENTITIES
  EVERY     DEPOT          OWNS A  REPAIRED.SET
  EVERY     BASE            HAS A  BASE.MISSILE.COUNTER,
                                A  BASE.MISSILE.RQMT
           THE SYSTEM      OWNS AN ALERT.MISSILE.SET,
                                A  PRODUCTION.POOL,
                                A  TAKE.SET,
                                A  TEST.SET
''**************************************************
GENERATE LIST ROUTINES
TEMPORARY ENTITIES
  EVERY       ENGINE          HAS AN ARRIVAL.TIME,
                                  A  CLAIM,
                                  AN ENGINE.ID.NUMBER,
                                  AN DATE.EXPIRES,
                                  A  LOCATION,
                                  A  RANK.DATE,
                                  A  SHIPPING.STATUS,
                                  A  START.DATE,
                                  A  SPARE.STATUS,
                                  A  STATUS,
                                  A  TEST.STATUS,
                                  A  TYPE,
                                  A  TYPE.SERVICE AND
              MAY BELONG TO       AN ALERT.MISSILE.SET,
                                  A  PRODUCTION.POOL,
                                  A  REPAIRED.SET,
                                  A  TAKE.SET,
                                  A  TEST.SET
INHIBIT LIST ROUTINES
''**************************************************
DEFINE ALERT.MISSILE.SET AS A SET RANKED BY
LOW RANK.DATE
DEFINE TAKE.SET  AS A SET RANKED BY HIGH STATUS,THEN BY
LOW DATE.EXPIRES
```

```
'' PREAMBLE CONTINUED
''***************************************************
RESOURCES INCLUDE WORK.STATION
''***************************************************
DEFINE LOC.REPAIRED.SET1         TO MEAN 1
DEFINE LOC.PRODUCTION            TO MEAN 9
DEFINE YES                       TO MEAN 10
DEFINE INTRANSIT                 TO MEAN 20
DEFINE BOEING                    TO MEAN 30
DEFINE OTL.TEST.SITE             TO MEAN 40
DEFINE EVIP.TEST.SITE            TO MEAN 50
DEFINE TAKEN                     TO MEAN 70
DEFINE LIMITED                   TO MEAN 100
DEFINE FULL                      TO MEAN 200
DEFINE REPAIR                    TO MEAN 300
DEFINE REFURB                    TO MEAN 400
DEFINE CONVERSION                TO MEAN 500
DEFINE NONE                      TO MEAN 600
DEFINE DUE                       TO MEAN 610
DEFINE BROKE                     TO MEAN 630
DEFINE SPARE                     TO MEAN 650
DEFINE OTL                       TO MEAN 700
DEFINE EVIP                      TO MEAN 800
DEFINE JTA                       TO MEAN 900
''***************************************************
DEFINE AVERAGE.AGE   AS AN INTEGER, 2-DIMENSIONAL ARRAY
DEFINE MAINTENANCE.RECORD
                     AS AN INTEGER, 2-DIMENSIONAL ARRAY
DEFINE WORK.STATIONS.IN.USE
                     AS AN INTEGER, 2-DIMENSIONAL ARRAY
DEFINE SPARES        AS AN INTEGER, 2-DIMENSIONAL ARRAY
DEFINE TERM          AS AN INTEGER, 1-DIMENSIONAL ARRAY
DEFINE MAY.CONVERT   AS AN INTEGER, 1-DIMENSIONAL ARRAY
''***************************************************
DEFINE START.DAY,  START.MO,  START.YR,
       END.DAY,  END.MO,  END.YR,
       OPEN.DAY, OPEN.MO,  OPEN.YR,
       CONVERT.DAY,  CONVERT.MO,  CONVERT.YR,
       DAY.CONVERSIONS.END, MO.CONVERSIONS.END,
       YR.CONVERSIONS.END,
       STREAM1,  STREAM2,  STREAM3,  STREAM4,  STREAM5
       AS INTEGER VARIABLES
DEFINE CONVERSION.TIME, FULL.TIME, LIMITED.TIME,
       REFURB.TIME,
       NUMBER.BASES, NUMBER.SPARES,
       OCALC.CAPACITY, WILLIAMS.CAPACITY,
       TOTAL.MONTHS, TRANSPORT.DAYS,
       EVIP.TEST,   OTL.TEST,   DESTROYED,   COUNT,
       NUM.BROKEN, NUM.OVERDUE, NUM.SPARE,
       ENGINE.SEQUENCE.NUMBER,  EARLY.OVERHAUL.OK
       AS INTEGER VARIABLES
DEFINE RECOVERY.FAILURE.RATE, START.DATE, FAIL.RATE,
       DATE.EXPIRES,RANK.DATE       AS REAL    VARIABLES
```

```
'' PREAMBLE CONTINUED
''*****************************************************
TALLY
         MAX.PER.MONTH AS THE MONTHLY MAXIMUM,
         LIFE.MAX      AS THE         MAXIMUM
                       OF             NUM.SPARE
ACCUMULATE
         BROKENUM      AS THE         NUMBER,
         BROKESUM      AS THE         SUM
                       OF             NUM.BROKEN
ACCUMULATE
         OVERNUM       AS THE         NUMBER,
         OVERSUM       AS THE         SUM
                       OF             NUM.OVERDUE
END
''*****************************************************




MAIN
DEFINE I,J AS INTEGER VARIABLES
DEFINE RZ,INFILE AS TEXT VARIABLES
'' RZ IS USED TO ACCOUNT FOR INPUT FILE COMMENTS
USE 7 FOR INPUT      USE 8 FOR OUTPUT
READ INFILE
PRINT 1 LINE WITH INFILE THUS
SOURCE FILE IS ****
LET ENGINE.SEQUENCE.NUMBER = 1
RESERVE TERM(*) AS 4        RESERVE MAY.CONVERT(*) AS 2
READ  TRANSPORT.DAYS,RZ,OTL.TEST,RZ,EVIP.TEST,RZ,
      RECOVERY.FAILURE.RATE,RZ
READ  REFURB.TIME,RZ
READ  FULL.TIME,RZ,LIMITED.TIME,RZ,CONVERSION.TIME,RZ,
      FAIL.RATE,RZ
USE 9 FOR INPUT
READ  START.MO,RZ,START.DAY,RZ,START.YR,RZ
READ  DAY.CONVERSIONS.END,RZ,MO.CONVERSIONS.END,RZ,
      YR.CONVERSIONS.END,RZ
READ  END.MO,RZ,END.DAY,RZ,END.YR,RZ,OPEN.MO,RZ,
      OPEN.DAY,RZ
READ  OPEN.YR,RZ,CONVERT.MO,RZ,CONVERT.DAY,RZ,
      CONVERT.YR,RZ
READ  STREAM1,RZ,STREAM2,RZ,STREAM3,RZ,STREAM4,RZ,
      STREAM5,RZ
READ  TERM(1),RZ,TERM(4),RZ,NUMBER.BASES,RZ,
      NUMBER.SPARES,RZ
LET TOTAL.MONTHS  = ((END.YR-START.YR)+1)*12+
    (END.MO-START.MO)+1
RESERVE MAINTENANCE.RECORD(*,*) AS TOTAL.MONTHS BY 5
RESERVE AVERAGE.AGE(*,*) AS TOTAL.MONTHS BY 2
RESERVE SPARES(*,*)  AS TOTAL.MONTHS BY 2
RESERVE WORK.STATIONS.IN.USE(*,*) AS TOTAL.MONTHS BY 2
```

```
CREATE EVERY WORK.STATION(2)   CREATE EVERY DEPOT(2)
CREATE EVERY BASE(NUMBER.BASES)
FOR EACH BASE
  READ BASE.MISSILE.RQMT(BASE),RZ
READ  WILLIAMS.CAPACITY,RZ,OCALC.CAPACITY,RZ,
      EARLY.OVERHAUL.OK,RZ
LET U.WORK.STATION(1) = WILLIAMS.CAPACITY
LET U.WORK.STATION(2) = OCALC.CAPACITY
CALL ORIGIN.R(START.MO,START.DAY,START.YR)
FOR I = START.YR TO END.YR
DO FOR J = 1 TO 12
  ACTIVATE A MONTHLY.STATISTICS AT DATE.F(J,1,I)
LOOP
ACTIVATE A HALT AT DATE.F(END.MO,END.DAY,END.YR)
SCHEDULE A SPARE.ENGINE.GENERATION NOW
ACTIVATE A TEST.GENERATION NOW
ACTIVATE A MISSILE.PRODUCTION NOW
ACTIVATE AN ENGINE.PRODUCTION NOW
ACTIVATE A SCHEDULE.CONVERSION NOW
SCHEDULE A DEPOT1.CAPACITY.SCHEDULE NOW
SCHEDULE A DEPOT2.CAPACITY.SCHEDULE NOW
START SIMULATION
END
```

```
PROCESS MAINTENANCE(MSL,DEPOT.NUMBER)
DEFINE MSL,DEPOT.NUMBER,INDEX AS INTEGER VARIABLES
LET SHIPPING.STATUS(MSL) = NONE
LET INDEX = TYPE.SERVICE(MSL)/100
ADD 1 TO MAINTENANCE.RECORD(COUNT,INDEX)
IF INDEX NE 1
  LET INDEX = 2
ALWAYS
IF TIME.V >= DATE.F(CONVERT.MO,CONVERT.DAY,CONVERT.YR)
AND TYPE(MSL) = 101 AND MAY.CONVERT(INDEX) > 0
    LET TYPE.SERVICE(MSL) = CONVERSION
    LET DEPOT.NUMBER = 1
    SUBTRACT 1 FROM MAY.CONVERT(INDEX)
ALWAYS
REQUEST 1 WORK.STATION(DEPOT.NUMBER)
ADD 1 TO WORK.STATIONS.IN.USE(COUNT,DEPOT.NUMBER)
IF TYPE.SERVICE(MSL) = FULL OR TYPE.SERVICE(MSL) =REPAI
  WORK FULL.TIME DAYS
  LET TYPE.SERVICE(MSL) = LIMITED
ELSE
  IF TYPE.SERVICE(MSL) = LIMITED
    WORK LIMITED.TIME DAYS
    LET TYPE.SERVICE(MSL) = FULL
  ELSE
    IF TYPE.SERVICE(MSL) = CONVERSION
      ADD 1 TO MAINTENANCE.RECORD(COUNT,5)
      WORK CONVERSION.TIME DAYS
      LET TYPE(MSL) = 104
      LET TYPE.SERVICE(MSL) = LIMITED
    ELSE
      WORK REFURB.TIME DAYS
      LET TYPE.SERVICE(MSL) = LIMITED
    ALWAYS
  ALWAYS
ALWAYS
RELINQUISH 1 WORK.STATION(DEPOT.NUMBER)
CALL DATE(MSL)
CALL SUBTRACT.FROM.SPARE.COUNT GIVING MSL
IF LOCATION(MSL) = LOC.PRODUCTION
  CALL SHIP GIVING MSL
  CALL SUBTRACT.FROM.SPARE.COUNT GIVING MSL
  WAIT TRANSPORT.DAYS DAYS
LET SHIPPING.STATUS(MSL) = NONE
  FILE THIS MSL IN THE PRODUCTION.POOL
  RETURN
ALWAYS
```

```
'' PROCESS MAINTENANCE CONTINUED
IF TEST.STATUS(MSL) = OTL
  LET TEST.STATUS(MSL) = NONE
  CALL SHIP GIVING MSL
  SCHEDULE A DEPLOY(MSL,LOCATION(MSL)) IN
  TRANSPORT.DAYS DAYS
  LET LOCATION(MSL) = LOC.PRODUCTION
ELSE
  LET TEST.STATUS(MSL) = NONE
  LET LOCATION(MSL) = DEPOT.NUMBER
  IF N.REPAIRED.SET(1) + N.REPAIRED.SET(2) = 0
    IF N.TAKE.SET > 0 OR EARLY.OVERHAUL.OK = YES
      SCHEDULE AN IN.ACTION(MSL) NOW
    ELSE
      GO HERE
    ALWAYS
  ELSE
    'HERE'
    FILE THIS  MSL IN REPAIRED.SET(DEPOT.NUMBER)
    ACTIVATE AN IN.ACTION NOW
  ALWAYS
ALWAYS
END
```

```
PROCESS SPARE.ENGINE.GENERATION
DEFINE I,J,K,L AS INTEGER VARIABLES
FOR I = 1982 TO 1983
DO FOR J = 1 TO 12
  DO FOR K = 1 TO 5
    DO
      IF L = NUMBER.SPARES
       RETURN
      ALWAYS
      WAIT DATE.F(J,K*5+RANDI.F(-2,2,STREAM4),I)-
      TIME.V DAYS
      CREATE AN ENGINE
      LET TYPE.SERVICE(ENGINE) = LIMITED
      LET TYPE(ENGINE) = 101
      LET ENGINE.ID.NUMBER(ENGINE) =
      ENGINE.SEQUENCE.NUMBER
      LET TEST.STATUS(ENGINE) = NONE
      ADD 1 TO ENGINE.SEQUENCE.NUMBER
      CALL DATE(ENGINE)
      LET LOCATION(ENGINE) = LOC.REPAIRED.SET1
      IF N.REPAIRED.SET(1) + N.REPAIRED.SET(2) = 0
        IF N.TAKE.SET > 0 OR EARLY.OVERHAUL.OK = YES
          SCHEDULE AN IN.ACTION(ENGINE) NOW
        ELSE
          GO HERE
        ALWAYS
      ELSE
        'HERE'
        FILE THIS  ENGINE IN REPAIRED.SET(1)
        ACTIVATE AN IN.ACTION NOW
      ALWAYS
      ADD 1 TO L
    LOOP
  LOOP
LOOP
END


PROCESS TEST.GENERATION
DEFINE I, NUMBER.TESTS,TEST.DAYS,TEST.TYPE AS
INTEGER VARIABLES
USE 11 FOR INPUT
READ NUMBER.TESTS
FOR I = 1 TO NUMBER.TESTS
DO
  READ TEST.DAYS,TEST.TYPE
  ACTIVATE A TEST.PICK(TEST.TYPE) IN TEST.DAYS DAYS
LOOP
END
```

```
PROCESS TEST.PICK(TYPETEST)
DEFINE TYPETEST,PICK AS INTEGER VARIABLES
FOR EACH ENGINE OF ALERT.MISSILE.SET,
WITH TEST.STATUS(ENGINE) = NONE
AND CLAIM(ENGINE) NE TAKEN, FIND PICK = ENGINE
LET TEST.STATUS(PICK) = TYPETEST
IF TYPETEST = EVIP
  IF N.REPAIRED.SET(1) + N.REPAIRED.SET(2) > 0 AND
  N.TAKE.SET = 0
    SCHEDULE AN IN.ACTION(0,PICK) NOW
  ELSE
    IF PICK IS NOT IN TAKE.SET
      FILE THIS PICK IN THE TAKE.SET
      ACTIVATE AN IN.ACTION NOW
    ALWAYS
  ALWAYS
  CALL ADD.TO.SPARE.COUNT GIVING PICK
ELSE
  IF STATUS(PICK) = DUE
    SUBTRACT 1 FROM NUM.OVERDUE
    LET STATUS(PICK) = NONE
  ALWAYS
  REMOVE THIS PICK FROM THE ALERT.MISSILE.SET
  IF PICK IS IN TAKE.SET
    REMOVE THIS PICK FROM TAKE.SET
  ALWAYS
  IF TYPETEST = OTL
    CALL SHIP GIVING PICK
    ACTIVATE A TEST(PICK) IN TRANSPORT.DAYS DAYS
  ELSE  '' JTA
    ADD 1 TO DESTROYED
  ALWAYS
ALWAYS
END
```

```
PROCESS MISSILE.PRODUCTION
DEFINE RZ AS A TEXT VARIABLE
DEFINE I,J,K,NUMBER.TO.SEND,MSL.PRODUCTION.NUMBER,
        DAY AS INTEGER VARIABLES
USE 13 FOR INPUT
READ RZ
FOR I = 1981 TO 1986
DO
  READ RZ
  FOR J = 1 TO 12
  DO
    READ MSL.PRODUCTION.NUMBER
    IF MSL.PRODUCTION.NUMBER = 0  CYCLE  ALWAYS
    LET NUMBER.TO.SEND =TRUNC.F(MSL.PRODUCTION.NUMBER/4)
    FOR K = 1 TO 3
    DO
      LET DAY = (K*7)+RANDI.F(-2,2,STREAM4)
      SCHEDULE A SHIP.MISSILE(NUMBER.TO.SEND) AT
      DATE.F(J,DAY,I)
    LOOP
    SCHEDULE A SHIP.MISSILE(MSL.PRODUCTION.NUMBER-
    NUMBER.TO.SEND*3) AT DATE.F(J,28,I)
  LOOP
LOOP
END



PROCESS SHIP.MISSILE(NUMBER)
DEFINE I,NUMBER,ENGINE AS INTEGER VARIABLES
DEFINE J AS A SAVED INTEGER VARIABLE
FOR I = 1 TO NUMBER
DO
'TRY.AGAIN'
  IF PRODUCTION.POOL IS NOT  EMPTY
    REMOVE FIRST ENGINE FROM PRODUCTION.POOL
  ELSE
    WAIT 1 DAY
    GO TRY.AGAIN
  ALWAYS
'CK'
  IF BASE.MISSILE.RQMT(J+1) > BASE.MISSILE.COUNTER(J+1)
  CALL SHIP GIVING ENGINE
    SCHEDULE A DEPLOY(ENGINE,J+1) IN TRANSPORT.DAYS DAYS
    ADD 1 TO BASE.MISSILE.COUNTER(J+1)
  ELSE
    ADD 1 TO J
    GO CK
  ALWAYS
LOOP
END


```

```
PROCESS ENGINE.PRODUCTION
DEFINE RZ AS A TEXT VARIABLE
DEFINE I,J,K,NUMBER.OF.ENGINES.TO.PRODUCE AS
 INTEGER VARIABLES
DEFINE DATE.TO.MAKE.ENGINE AS A REAL VARIABLE
USE 15 FOR INPUT
READ RZ
FOR I = 1981 TO 1986
DO
  READ RZ
  FOR J = 1 TO 12
  DO
    READ NUMBER.OF.ENGINES.TO.PRODUCE
    IF NUMBER.OF.ENGINES.TO.PRODUCE = 0  CYCLE  ALWAYS
    LET DATE.TO.MAKE.ENGINE =
    30/NUMBER.OF.ENGINES.TO.PRODUCE
    FOR K = 1 TO NUMBER.OF.ENGINES.TO.PRODUCE
    DO
      ACTIVATE A CREATE.ENGINE IN DATE.F(J,1,I)+
      DATE.TO.MAKE.ENGINE DAYS
      ADD 30/NUMBER.OF.ENGINES.TO.PRODUCE TO
      DATE.TO.MAKE.ENGINE
    LOOP
  LOOP
LOOP
END



PROCESS CREATE.ENGINE
CREATE AN ENGINE
LET ENGINE.ID.NUMBER(ENGINE) = ENGINE.SEQUENCE.NUMBER
LET TEST.STATUS(ENGINE) = NONE
LET TYPE(ENGINE) = 101
ADD 1 TO ENGINE.SEQUENCE.NUMBER
LET TYPE.SERVICE(ENGINE) = LIMITED
CALL DATE(ENGINE)
LET LOCATION(ENGINE) = LOC.PRODUCTION
FILE THIS ENGINE IN THE PRODUCTION.POOL
END
```

```
ROUTINE DATE(ENGINE)
DEFINE ENGINE AS AN INTEGER VARIABLE
LET START.DATE(ENGINE) = INT.F(TIME.V)
IF RANDOM.F(STREAM2) < FAIL.RATE
  LET DATE.EXPIRES(ENGINE) = TIME.V +
  RANDOM.F(STREAM3)*TERM(TYPE(ENGINE)-100)
  LET TYPE.SERVICE(ENGINE) = REPAIR
  ACTIVATE A FAILURE.ACTION(ENGINE) AT
  DATE.EXPIRES(ENGINE)
ELSE
  LET DATE.EXPIRES(ENGINE) = TIME.V +
  TERM(TYPE(ENGINE)-100)
  ACTIVATE AN OVERDUE.ACTION(ENGINE) AT
  DATE.EXPIRES(ENGINE) - TRANSPORT.DAYS
ALWAYS
LET RANK.DATE(ENGINE) = TERM(TYPE(ENGINE)-100) + TIME.V
END


PROCESS SCHEDULE.CONVERSION
DEFINE I,J AS INTEGER VARIABLES
DEFINE RZ AS A TEXT VARIABLE
USE 17 FOR INPUT
READ RZ
FOR I = 1988 TO 1991
DO FOR J = 1 TO 12
  ACTIVATE A CONVERT AT DATE.F(J,1,I)
LOOP
END



PROCESS CONVERT
DEFINE LIMITED.QUOTA,FULL.QUOTA AS INTEGER VARIABLES
USE 17 FOR INPUT
READ LIMITED.QUOTA,FULL.QUOTA
ADD LIMITED.QUOTA TO MAY.CONVERT(1)
ADD FULL.QUOTA TO MAY.CONVERT(2)
END


PROCESS TRANSPORT(MSL)
DEFINE MSL AS AN INTEGER VARIABLE
CALL ADD.TO.SPARE.COUNT GIVING MSL
IF TYPE.SERVICE(MSL) NE LIMITED
OR TIME.V < DATE.F(OPEN.MO,OPEN.DAY,OPEN.YR)
OR N.X.WORK.STATION(1)/WILLIAMS.CAPACITY <
N.X.WORK.STATION(2)/OCALC.CAPACITY
  SCHEDULE A MAINTENANCE(MSL,1) IN TRANSPORT.DAYS DAYS
ELSE
  SCHEDULE A MAINTENANCE(MSL,2) IN TRANSPORT.DAYS DAYS
ALWAYS
CALL SHIP GIVING MSL
END
```

```
PROCESS TEST(MSL)
DEFINE MSL AS AN INTEGER VARIABLE
LET SHIPPING.STATUS(MSL) = NONE
FILE THIS MSL IN THE TEST.SET
IF TEST.STATUS(MSL) = EVIP
  WAIT EVIP.TEST DAYS
ELSE
  WAIT OTL.TEST DAYS
ALWAYS
IF MSL IS IN TEST.SET
  REMOVE THIS MSL FROM TEST.SET
  IF TEST.STATUS(MSL) = OTL AND
  RANDOM.F(STREAM1) >= RECOVERY.FAILURE.RATE OR
  TEST.STATUS(MSL) = EVIP
    LET TYPE.SERVICE(MSL) = REFURB
    ACTIVATE A TRANSPORT(MSL) NOW
  ELSE
    ADD 1 TO DESTROYED
  ALWAYS
ALWAYS
END
```

```
PROCESS OVERDUE.ACTION(ENGINE)
DEFINE ENGINE AS AN INTEGER VARIABLE
IF DATE.EXPIRES(ENGINE) NE INT.F(TIME.V) +TRANSPORT.DAYS
  RETURN
ALWAYS
IF CLAIM(ENGINE) = TAKEN
  WAIT TRANSPORT.DAYS DAYS
ALWAYS
IF ENGINE IS IN ALERT.MISSILE.SET
  LET STATUS(ENGINE) = DUE
  ADD 1 TO NUM.OVERDUE
  CALL ADD.TO.SPARE.COUNT GIVING ENGINE
  IF N.TAKE.SET = 0 AND N.REPAIRED.SET(1) +
  N.REPAIRED.SET(2) > 0
    ACTIVATE AN IN.ACTION(0,ENGINE) NOW
  ELSE
    IF ENGINE IS NOT IN TAKE.SET
      FILE THIS ENGINE IN THE TAKE.SET
    ALWAYS
    ACTIVATE AN IN.ACTION NOW
  ALWAYS
ELSE
  IF ENGINE IS IN REPAIRED.SET
    CALL REMOVE.FROM.REPAIRED.SET(ENGINE)
    ACTIVATE A MAINTENANCE(ENGINE,1) NOW
  ELSE
    IF ENGINE IS IN PRODUCTION.POOL
      REMOVE THIS ENGINE FROM THE PRODUCTION.POOL
      CALL SHIP GIVING ENGINE
      ACTIVATE A MAINTENANCE(ENGINE,1) IN
      TRANSPORT.DAYS DAYS
    ELSE
      IF SHIPPING.STATUS(ENGINE) = INTRANSIT
        LET DATE.EXPIRES(ENGINE) = ARRIVAL.TIME(ENGINE)+
         1 + TRANSPORT.DAYS
        ACTIVATE AN OVERDUE.ACTION(ENGINE) AT
        ARRIVAL.TIME(ENGINE) + 1
      ALWAYS
    ALWAYS
  ALWAYS
ALWAYS
END


```

```
PROCESS FAILURE.ACTION(ENGINE)
DEFINE ENGINE AS AN INTEGER VARIABLE
IF DATE.EXPIRES(ENGINE) NE TIME.V
  RETURN
ALWAYS
IF CLAIM(ENGINE) = TAKEN
  WAIT TRANSPORT.DAYS
ALWAYS
IF ENGINE IS IN ALERT.MISSILE.SET
  LET STATUS(ENGINE) = BROKE
  ADD 1 TO NUM.BROKEN
  CALL ADD.TO.SPARE.COUNT GIVING ENGINE
  REMOVE THIS ENGINE FROM THE ALERT.MISSILE.SET
  IF N.TAKE.SET = 0 AND N.REPAIRED.SET(1) +
  N.REPAIRED.SET(2) > 0
    ACTIVATE AN IN.ACTION(0,ENGINE) NOW
  ELSE
    IF ENGINE IS NOT IN TAKE.SET
      FILE THIS ENGINE IN THE TAKE.SET
    ALWAYS
    ACTIVATE AN IN.ACTION NOW
  ALWAYS
ELSE
  IF ENGINE IS IN REPAIRED.SET
    CALL REMOVE.FROM.REPAIRED.SET(ENGINE)
    ACTIVATE A MAINTENANCE(ENGINE,1) NOW
  ELSE
    IF ENGINE IS IN PRODUCTION.POOL
      REMOVE THIS ENGINE FROM THE PRODUCTION.POOL
      CALL SHIP GIVING ENGINE
      ACTIVATE A MAINTENANCE(ENGINE,1) IN
      TRANSPORT.DAYS DAYS
    ELSE
      IF ENGINE IS IN TEST.SET
        REMOVE THIS ENGINE FROM TEST.SET
        CALL SHIP GIVING ENGINE
        ACTIVATE A MAINTENANCE(ENGINE,1) IN
        TRANSPORT.DAYS DAYS
      ELSE
        IF SHIPPING.STATUS(ENGINE) = INTRANSIT
          LET DATE.EXPIRES(ENGINE) =
          ARRIVAL.TIME(ENGINE) + 1 + TRANSPORT.DAYS
          ACTIVATE A FAILURE.ACTION(ENGINE) AT
          ARRIVAL.TIME(ENGINE) +1
        ALWAYS
      ALWAYS
    ALWAYS
  ALWAYS
ALWAYS
END
```

```
PROCESS IN.ACTION(MSL,FLAG)
DEFINE MSL,FLAG AS INTEGER VARIABLES
IF FLAG > 100
  GO OUT
ALWAYS
FOR EACH ENGINE OF TAKE.SET,WITH CLAIM(ENGINE) NE TAKEN,
FIND FLAG= ENGINE,
IF FOUND
  GO OUT
ALWAYS
IF EARLY.OVERHAUL.OK = YES
  FOR EACH ENGINE OF ALERT.MISSILE.SET, WITH
  CLAIM(ENGINE) NE TAKEN AND STATUS(ENGINE) = NONE,
  FIND FLAG = ENGINE,
  IF FOUND
    GO OUT
  ALWAYS
ALWAYS
IF MSL NE 0
  IF MSL IS NOT IN REPAIRED.SET
    FILE THIS MSL IN REPAIRED.SET(1)
  ALWAYS
ALWAYS
RETURN
'OUT'
IF MSL > 100
  GO OUT1
ALWAYS
IF N.REPAIRED.SET(1) IS >= N.REPAIRED.SET(2)
AND REPAIRED.SET(1) IS NOT EMPTY
  REMOVE FIRST MSL FROM REPAIRED.SET(1)
ELSE
  IF REPAIRED.SET(2) IS NOT EMPTY
    REMOVE FIRST MSL FROM REPAIRED.SET(2)
  ELSE
    IF FLAG NOT IN TAKE.SET AND STATUS(FLAG) NE NONE
      FILE THIS FLAG IN TAKE.SET
    ALWAYS
    RETURN
  ALWAYS
ALWAYS
'OUT1'
CALL SHIP GIVING MSL
LET CLAIM(FLAG) = TAKEN
ACTIVATE AN ENTER.BASE(MSL,FLAG) IN TRANSPORT.DAYS DAYS
END
```

```
PROCESS ENTER.BASE(ENGINE.IN,FLAG)
DEFINE ENGINE.IN,ENGINE.OUT,FLAG AS INTEGER VARIABLES
LET SHIPPING.STATUS(ENGINE.IN) = NONE
IF  N.TAKE.SET = 0
  LET ENGINE.OUT = FLAG
  LET CLAIM(ENGINE.OUT) = NONE
  GO START
ALWAYS
LET CLAIM(FLAG) = NONE
IF FLAG IS NOT IN TAKE.SET AND STATUS(FLAG) NE NONE
  FILE THIS FLAG IN TAKE.SET
ALWAYS
FOR EACH ENGINE OF TAKE.SET, WITH LOCATION(ENGINE) =
LOCATION(FLAG) AND CLAIM(ENGINE) NE TAKEN,
FIND ENGINE.OUT = ENGINE,
IF FOUND
  GO START
ALWAYS
LET ENGINE.OUT = FLAG
'START'
IF TEST.STATUS(ENGINE.OUT) = EVIP AND STATUS(ENGINE.OUT)
NE BROKE
  CALL SHIP GIVING ENGINE.OUT
  ACTIVATE A TEST(ENGINE.OUT) IN TRANSPORT.DAYS DAYS
ELSE
  ACTIVATE A TRANSPORT(ENGINE.OUT) NOW
ALWAYS
IF STATUS(ENGINE.OUT) = DUE
  SUBTRACT 1 FROM NUM.OVERDUE
ELSE
  IF STATUS(ENGINE.OUT) = BROKE
    SUBTRACT 1 FROM NUM.BROKEN
  ALWAYS
ALWAYS
'SWAP'
IF ENGINE.OUT IS IN TAKE.SET
  REMOVE THIS ENGINE.OUT FROM TAKE.SET
ALWAYS
IF M.ALERT.MISSILE.SET(ENGINE.OUT) NE 0
  REMOVE THIS ENGINE.OUT FROM ALERT.MISSILE.SET
ALWAYS
LET STATUS(ENGINE.OUT) = NONE
LET LOCATION(ENGINE.IN) = LOCATION(ENGINE.OUT)
FILE THIS ENGINE.IN IN THE ALERT.MISSILE.SET
END
```

```
PROCESS DEPOT1.CAPACITY.SCHEDULE
DEFINE MONTH,DAY,YEAR,CAPACITY,OLD AS INTEGER VARIABLES
LET OLD = 200
'START'
USE 19 FOR INPUT
IF DATA IS NOT ENDED
  READ CAPACITY,MONTH,DAY,YEAR
  IF DATE.F(MONTH,DAY,YEAR) >= TIME.V
    WAIT DATE.F(MONTH,DAY,YEAR) - TIME.V DAYS
    RELINQUISH (200-OLD) UNITS OF WORK.STATION(1)
    REQUEST (200-CAPACITY) UNITS OF WORK.STATION(1)
    WITH PRIORITY 1
    LET OLD = CAPACITY
  ALWAYS
  GO TO START
ALWAYS
END



PROCESS DEPOT2.CAPACITY.SCHEDULE
DEFINE MONTH,DAY,YEAR,CAPACITY,OLD AS INTEGER VARIABLES
LET OLD = 200
'START'
USE 21 FOR INPUT
IF DATA IS NOT ENDED
  READ CAPACITY,MONTH,DAY,YEAR
  IF DATE.F(MONTH,DAY,YEAR) >= TIME.V
    WAIT DATE.F(MONTH,DAY,YEAR) - TIME.V DAYS
    RELINQUISH (200-OLD) UNITS OF WORK.STATION(2)
    REQUEST (200-CAPACITY) UNITS OF WORK.STATION(2)
    WITH PRIORITY 1
    LET OLD = CAPACITY
  ALWAYS
  GO TO START
ALWAYS
END
```

```
PROCESS HALT
DEFINE I,B2,O2 AS INTEGER VARIABLES
START NEW PAGE
PRINT 2 LINES WITH LIFE.MAX THUS
THE MAXIMUM NUMBER OF SPARE ENGINES REQUIRED OVER THE
LIFE OF THE SYSTEM IS ****
START NEW PAGE
FOR I = 0 TO END.YR - START.YR
DO
IF FRAC.F(I/5) = 0.  START NEW PAGE  ALWAYS
PRINT 1 LINE WITH (I+START.YR) THUS

****JAN FEB MAR APR MAY JUN JUL AUG SEP OCT NOV DEC
PRINT 1 LINE WITH SPARES(I*12+1,1),SPARES(I*12+2,1),
                  SPARES(I*12+3,1),SPARES(I*12+4,1),
                  SPARES(I*12+5,1),SPARES(I*12+6,1),
                  SPARES(I*12+7,1),SPARES(I*12+8,1),
                  SPARES(I*12+9,1),SPARES(I*12+10,1),
                  SPARES(I*12+11,1),SPARES(I*12+12,1)
                  THUS

INC *** *** *** *** *** *** *** *** *** *** *** ***
PRINT 1 LINE WITH SPARES(I*12+1,2),SPARES(I*12+2,2),
                  SPARES(I*12+3,2),SPARES(I*12+4,2),
                  SPARES(I*12+5,2),SPARES(I*12+6,2),
                  SPARES(I*12+7,2),SPARES(I*12+8,2),
                  SPARES(I*12+9,2),SPARES(I*12+10,2),
                  SPARES(I*12+11,2),SPARES(I*12+12,2)
                  THUS
TOT.REQ
*** *** *** *** *** *** *** *** *** *** *** ***
PRINT 1 LINE WITH
           AVERAGE.AGE(I*12+1,1),AVERAGE.AGE(I*12+2,1),
           AVERAGE.AGE(I*12+3,1),AVERAGE.AGE(I*12+4,1),
           AVERAGE.AGE(I*12+5,1),AVERAGE.AGE(I*12+6,1),
           AVERAGE.AGE(I*12+7,1),AVERAGE.AGE(I*12+8,1),
           AVERAGE.AGE(I*12+9,1),AVERAGE.AGE(I*12+10,1),
           AVERAGE.AGE(I*12+11,1),AVERAGE.AGE(I*12+12,1)
           THUS
AGE101
*** *** *** *** *** *** *** *** *** *** *** ***
PRINT 1 LINE WITH
           AVERAGE.AGE(I*12+1,2),AVERAGE.AGE(I*12+2,2),
           AVERAGE.AGE(I*12+3,2),AVERAGE.AGE(I*12+4,2),
           AVERAGE.AGE(I*12+5,2),AVERAGE.AGE(I*12+6,2),
           AVERAGE.AGE(I*12+7,2),AVERAGE.AGE(I*12+8,2),
           AVERAGE.AGE(I*12+9,2),AVERAGE.AGE(I*12+10,2),
           AVERAGE.AGE(I*12+11,2),AVERAGE.AGE(I*12+12,2)
           THUS
AGE104
*** *** *** *** *** *** *** *** *** *** *** ***
```

```
    '' PROCESS HALT CONTINUED
797 PRINT 1 LINE WITH
     MAINTENANCE.RECORD(I*12+1,1),
     MAINTENANCE.RECORD(I*12+2,1),
799  MAINTENANCE.RECORD(I*12+3,1),
     MAINTENANCE.RECORD(I*12+4,1),
800  MAINTENANCE.RECORD(I*12+5,1),
     MAINTENANCE.RECORD(I*12+6,1),
801  MAINTENANCE.RECORD(I*12+7,1),
     MAINTENANCE.RECORD(I*12+8,1),
802  MAINTENANCE.RECORD(I*12+9,1),
     MAINTENANCE.RECORD(I*12+10,1),
803  MAINTENANCE.RECORD(I*12+11,1),
     MAINTENANCE.RECORD(I*12+12,1) THUS
804 MINORS
    *** *** *** *** *** *** *** *** *** *** *** ***
805 PRINT 1 LINE WITH
     MAINTENANCE.RECORD(I*12+1,2),
806  MAINTENANCE.RECORD(I*12+2,2),
807  MAINTENANCE.RECORD(I*12+3,2),
     MAINTENANCE.RECORD(I*12+4,2),
808  MAINTENANCE.RECORD(I*12+5,2),
     MAINTENANCE.RECORD(I*12+6,2),
809  MAINTENANCE.RECORD(I*12+7,2),
     MAINTENANCE.RECORD(I*12+8,2),
810  MAINTENANCE.RECORD(I*12+9,2),
     MAINTENANCE.RECORD(I*12+10,2),
811  MAINTENANCE.RECORD(I*12+11,2),
     MAINTENANCE.RECORD(I*12+12,2) THUS
812 MAJORS
    *** *** *** *** *** *** *** *** *** *** *** ***
813 PRINT 1 LINE WITH
     MAINTENANCE.RECORD(I*12+1,3),
814  MAINTENANCE.RECORD(I*12+2,3),
815  MAINTENANCE.RECORD(I*12+3,3),
     MAINTENANCE.RECORD(I*12+4,3),
816  MAINTENANCE.RECORD(I*12+5,3),
     MAINTENANCE.RECORD(I*12+6,3),
817  MAINTENANCE.RECORD(I*12+7,3),
     MAINTENANCE.RECORD(I*12+8,3),
818  MAINTENANCE.RECORD(I*12+9,3),
     MAINTENANCE.RECORD(I*12+10,3),
819  MAINTENANCE.RECORD(I*12+11,3),
     MAINTENANCE.RECORD(I*12+12,3) THUS
820 UNSCHD
    *** *** *** *** *** *** *** *** *** *** *** ***
```

```
'' PROCESS HALT CONTINUED
PRINT 1 LINE WITH
 MAINTENANCE.RECORD(I*12+1,4),
 MAINTENANCE.RECORD(I*12+2,4),
 MAINTENANCE.RECORD(I*12+3,4),
 MAINTENANCE.RECORD(I*12+4,4),
 MAINTENANCE.RECORD(I*12+5,4),
 MAINTENANCE.RECORD(I*12+6,4),
 MAINTENANCE.RECORD(I*12+7,4),
 MAINTENANCE.RECORD(I*12+8,4),
 MAINTENANCE.RECORD(I*12+9,4),
 MAINTENANCE.RECORD(I*12+10,4),
 MAINTENANCE.RECORD(I*12+11,4),
 MAINTENANCE.RECORD(I*12+12,4) THUS
REFURB
*** *** *** *** *** *** *** *** *** *** *** ***
PRINT 2 LINES WITH
 MAINTENANCE.RECORD(I*12+1,5),
 MAINTENANCE.RECORD(I*12+2,5),
 MAINTENANCE.RECORD(I*12+3,5),
 MAINTENANCE.RECORD(I*12+4,5),
 MAINTENANCE.RECORD(I*12+5,5),
 MAINTENANCE.RECORD(I*12+6,5),
 MAINTENANCE.RECORD(I*12+7,5),
 MAINTENANCE.RECORD(I*12+8,5),
 MAINTENANCE.RECORD(I*12+9,5),
 MAINTENANCE.RECORD(I*12+10,5),
 MAINTENANCE.RECORD(I*12+11,5),
 MAINTENANCE.RECORD(I*12+12,5) THUS
CONVER
*** *** *** *** *** *** *** *** *** *** *** ***

  LOOP
LET B2 = TRUNC.F(BROKENUM/2)
LET O2 = TRUNC.F(OVERNUM/2)
PRINT 5 LINES WITH B2,BROKESUM/B2,BROKESUM THUS
****** MISSILES WERE BROKEN AT DEPLOYMENT BASES THROUGH-
OUT THE LIFE OF THE SYSTEM. THEY  AVERAGED  ***.**  DAYS
ON BASE (BROKEN) BEFORE BEING REPLACED  FOR  A  TOTAL OF
******.** DAYS IN INOPERABLE STATUS.


PRINT 4 LINES WITH O2,  OVERSUM/O2 - TRANSPORT.DAYS,
   OVERSUM - O2 * TRANSPORT.DAYS THUS
 *****  MISSILES  WERE  KEPT  IN  A DEPLOYED STATUS PAST
 THEIR WARRANTED LIFE THROUGHOUT THE LIFE OF THE SYSTEM.
 THEY  AVERAGED  ***.**  DAYS ON  BASE (OVERDUE)  BEFORE
 BEING REPLACED FOR A TOTAL OF  ******.**  DAYS OVERDUE.
STOP
END
```

```
PROCESS MONTHLY.STATISTICS
DEFINE LAST1 AS A SAVED INTEGER VARIABLE
DEFINE DIVISOR.1,DIVISOR.4 AS INTEGER VARIABLES
DEFINE TOTAL.1, TOTAL.4 AS REAL VARIABLES
ADD 1 TO COUNT
LET DIVISOR.1 = 0    LET DIVISOR.4 = 0
LET TOTAL.1 = 0    LET TOTAL.4 = 0
FOR EACH ENGINE OF ALERT.MISSILE.SET
DO
  IF TYPE(ENGINE) = 101
    ADD 1 TO DIVISOR.1
    ADD TIME.V-START.DATE(ENGINE) TO TOTAL.1
  ELSE '' 104
    ADD 1 TO DIVISOR.4
    ADD TIME.V - START.DATE(ENGINE) TO TOTAL.4
  ALWAYS
LOOP
IF DIVISOR.1 = 0    LET DIVISOR.1 = 1    ALWAYS
IF DIVISOR.4 = 0    LET DIVISOR.4 = 1    ALWAYS
LET AVERAGE.AGE(COUNT,1) = TRUNC.F(TOTAL.1/DIVISOR.1)
LET AVERAGE.AGE(COUNT,2) = TRUNC.F(TOTAL.4/DIVISOR.4)
LET SPARES(COUNT,2) = MAX.PER.MONTH
IF LIFE.MAX - LAST1 > 0
  LET SPARES(COUNT,1) = LIFE.MAX - LAST1
  LET LAST1 = LIFE.MAX
ALWAYS
RESET MONTHLY TOTALS OF NUM.SPARE
END


ROUTINE SHIP(MSL)
LET ARRIVAL.TIME(MSL) = TRUNC.F(TIME.V) + TRANSPORT.DAYS
LET SHIPP.STATUS(MSL) = INTRANSIT
RETURN
END


PROCESS DEPLOY(MSL,BASE)
DEFINE MSL,BASE AS INTEGER VARIABLES
LET LOCATION(MSL) = BASE + 2
LET SHIPPING.STATUS(MSL) = NONE
FILE THIS MSL IN THE ALERT.MISSILE.SET
END



ROUTINE ADD.TO.SPARE.COUNT(MSL)
IF SPARE.STATUS(MSL) NE SPARE
  ADD 1 TO NUM.SPARE
  LET SPARE.STATUS(MSL) = SPARE
ALWAYS
RETURN
END
```

```
ROUTINE SUBTRACT.FROM.SPARE.COUNT(MSL)
IF SPARE.STATUS(MSL) = SPARE
  SUBTRACT 1 FROM NUM.SPARE
  LET SPARE.STATUS(MSL) = NONE
ALWAYS
RETURN
END



ROUTINE REMOVE.FROM.REPAIRED.SET(MSL)
DEFINE MSL, ENGINE AS INTEGER VARIABLES
FOR EACH ENGINE OF REPAIRED.SET(1),
WITH ENGINE.ID.NUMBER(ENGINE) = ENGINE.ID.NUMBER(MSL),
FIND THE FIRST CASE,
IF FOUND
  REMOVE THIS ENGINE FROM REPAIRED.SET(1)
  RETURN
ALWAYS
FOR EACH ENGINE OF REPAIRED.SET(2),
WITH ENGINE.ID.NUMBER(ENGINE) = ENGINE.ID.NUMBER(MSL),
FIND THE FIRST CASE,
IF FOUND
  REMOVE THIS ENGINE FROM REPAIRED.SET(2)
  RETURN
ALWAYS
END
```

END OF PROGRAM LISTING

Appendix F: Explanation of Program Terms

Refer to APPENDIX E: Program Listing for the context in which the following terms are used. Words appearing in all CAPITALS are names used in the program.

Arrays

AVERAGE.AGE - Stores the average age of all engines on alert by type, 101 or 104.

MAINTENANCE.RECORD - Stores the number of each type of depot service (FULL, LIMITED, etc.) accomplished at each depot.

MAY.CONVERT - The cumulative number of conversions scheduled per month for both limited and full service overhauls.

SPARES - Stores the maximum number of spare engines required by the system for each month.

TERM - Stores the length of warranty for each type of engine.

WORK.STATIONS.IN.USE - Stores the total number of WORK.STATIONs in use at each depot at any given time.

Attributes of Each Engine

ARRIVAL.TIME - The date an engine should arrive at the next process after being shipped.

CLAIM - Identifies an engine as being previously selected for a process, such as a test or exchange.

DATE.EXPIRES - The end of an engine's warranted lifetime or date of random failure.

ENGINE.ID.NUMBER - A unique identification for each engine.

LOCATION - Indicates where the engine is: at a base, depot, production, etc.

RANK.DATE - Date used to establish an engine's priority over other engines for selection for testing or early overhaul.

SHIPPING.STATUS - Indicates if the engine is INTRANSIT (being shipped from one location to another) or not.

START.DATE - Date engine's current warranted period begins.

SPARE.STATUS - Indicates if an engine is being used as a spare engine (while it is being processed in MAINTENANCE, TEST, TRANSPORT, etc).

STATUS - Shows if an engine is serviceable (NONE), requires servicing (DUE), or has failed (BROKE).

TEST.STATUS - Indicates the type of test an engine will undergo (NONE if not selected for testing).

TYPE - Indicates what model the engine is, F107-101 (101) or F107-104 (104).

TYPE.SERVICE - Shows what type of servicing is required next: FULL overhaul, LIMITED overhaul, REFURBishment after a test, REPAIR due to a failure, or CONVERSION if an engine will be converted from a 101 to a 104.

Permanent Entities

BASE - Six bases are in the original design with attributes of BASE.MISSILE.COUNTER and BASE.MISSILE.RQMT. These attributes are used to record the number of missiles actually at each base and the number required for each base.

DEPOT - Two depots, each with their own pool of repaired, serviceable engines called a REPAIRED.SET.

Processes

CONVERT - Reads input values from an external file (SIMU17) which are the quota of engines the system MAY.CONVERT each month during the conversion period.

CREATE.ENGINE - Creates new entities (engines) and assigns values to their attributes. Places the engines in the PRODUCTION.POOL.

DEPLOY - Updates an incoming missile's location to the correct base and places it in that base's ALERT.MISSILE.SET. Activated for a missile's initial deployment and for returning a REFURBished OTL test missile to its correct base.

DEPOT1.CAPACITY.SCHEDULE, DEPOT2.CAPACITY.SCHEDULE - Gives the program operator the capability to vary the number of WORK.STATIONs available at each depot. The number of stations can be adjusted on any date desired by reading the date and amount of capability from input files (SIMU19 and SIMU21).

ENGINE.PRODUCTION - Reads the NUMBER.OF.ENGINES.TO.PRODUCE from an input file (SIMU15) and activates the process CREATE.ENGINE to produce the designated number each month.

ENTER.BASE - Controls the entry and exit of engines to and from the base. Verifies the priority of selections made by IN.ACTION by reviewing the TAKE.SET. Activates processes depending on whether the outgoing engine is scheduled for a TEST or MAINTENANCE. Updates the number of engines broken or overdue as needed. Removes the outbound engine from base sets. Assigns the old engine's base location to the new engine and places the new engine in the ALERT.MISSILE.SET.

FAILURE.ACTION - Activated by the DATE routine. Processes engines which have failed. Reviews the ALERT.MISSILE.SET, TAKE.SET, or PRODUCTION.POOL as required to remove the failed engine from the set and schedules an IN.ACTION or MAINTENANCE as appropriate. Updates the engine's attributes to reflect this processing. It also changes system variables, such as NUM.BROKEN, to reflect a failure.

HALT - Prints the final report of the system's performance. Lists the spares used and each type of maintenance performed by year and month. Reports on the number of missiles broken in the simulation and the time it took to replace them. Reports the number and average time on alert, past the warranted time, for overdue engines.

IN.ACTION - Activated by a need for an engine or the availability of one. Tries to match a need with an available spare engine. Searches the TAKE.SET to see if it contains an engine needing to be changed out. If an engine at the base needs to be exchanged the REPAIRED.SETs are checked for the availability of spares. ENTER.BASE is activated if a match is found.

MAIN - Initiates program activity. Reads initial values for variables from input files SIMU7 and SIMU9. Dimensions the arrays and provides the appropriate number of DEPOTs, BASEs, and WORK.STATIONs. Schedules the processes that begin the simulation.

MAINTENANCE - Records the TYPE.SERVICE in the MAINTENANCE.RECORD. Determines if the engine should be converted and if the conversion service is available. Uses a WORK.STATION for the required amount of time and updates the

TYPE.SERVICE due for its next servicing. Releases the WORK.STATION and updates the engine's dates and SPARE.STATUS. Notifies the SYSTEM of a spare engine availability and schedules exchange processes as appropriate, or places the repaired engine in the depot stock.

MISSILE.PRODUCTION - Reads the missile production schedule from an input file (SIMU13). Schedules a SHIP.MISSILE to send the missiles to each base.

MONTHLY.STATISTICS - Computes and records the AVERAGE.AGE of the 101 and 104 type engines and various other system statistics. Records the number of SPARES used for the month and the total used for the life of the system.

OVERDUE.ACTION - Removes the engine from ALERT.MISSILE.SET, REPAIRED.SET, or PRODUCTION.POOL. Records the STATUS as DUE and schedules an IN.ACTION or MAINTENANCE as needed.

SCHEDULE.CONVERSION - Activates CONVERT processes each month during the conversion period.

SHIP.MISSILE - Selects engines for shipment from the PRODUCTION.POOL. Schedules the process DEPLOY until the BASE.MISSILE.COUNTER quantity for each base matches that base's BASE.MISSILE.RQMT.

SPARE.ENGINE.GENERATION - Produces the spare engines for the system and assigns initial values to their attributes. "Creates" engines at the rate of five per month until the NUMBER.SPARES to be created is reached. Places the engines in the REPAIRED.SET for depot 1 unless an immediate need is found, in which case the process IN.ACTION is activated to satisfy that need.

TEST - Places the engine in the TEST.SET while the test is in progress. The length of time in the TEST.SET depends on the type of test performed. After testing, the TYPE.SERVICE is set to REFURB with the exception of unrecovered OTL missiles, which are simply destroyed.

TEST.GENERATION - Reads the type and date of each test from an input file (SIMU11) and schedules the process TEST.PICK as required.

TEST.PICK - Selects, for each test, the engine from the ALERT.MISSILE.SET which is furthest past its warranty expiration date, or if none is past, the engine closest to its warranty expiration date. Engines picked for EVIP tests are processed through IN.ACTION to locate a spare engine to take their place at the base. The engine's STATUS and SPARE.STATUS are updated as needed. Missiles picked for OTL tests are scheduled for a TEST at this time.

TRANSPORT - Updates the SPARE.STATUS, then schedules the engine for MAINTENANCE at the appropriate depot.

Routines

ADD.TO.SPARE.COUNT - Records an increase in the number of spares needed (NUM.SPARES) and updates the SPARE.STATUS attribute of the engine.

DATE - Assigns a START.DATE, a DATE.EXPIRES, and a RANK.DATE to each engine. Schedules a FAILURE.ACTION or an OVERDUE.ACTION on the basis of a random number generated by the system. If this number is less than the FAIL.RATE, the engine will fail early. For a failed engine, the DATE.EXPIRES is a random percentage of the warranted lifetime.

REMOVE.FROM.REPAIRED.SET - Locates the desired engine in the REPAIRED.SET at one of the depots and removes it.

SUBTRACT.FROM.SPARE.COUNT - Records a decrease in the number of spares needed (NUM.SPARES) and updates the SPARE.STATUS attribute of the engine.

Sets

ALERT.MISSILE.SET - Contains missiles which are in operational use at a base.

PRODUCTION.POOL - Missiles are stored in this set until deployed to a base.

TAKE.SET - Maintains, in priority order, those engines which need to be exchanged but can't because a spare is not immediately available.

TEST.SET - Keeps track of engines during testing.

Variables

CONVERT.DAY, CONVERT.MO, CONVERT.YR - The date 101 to 104 conversions begin.

CONVERSION.TIME, FULL.TIME, LIMITED.TIME, REFURB.TIME - The amount of time required to complete depot maintenance for each type of service.

COUNT - A counter used as a subscript in various arrays to indicate which month of the simulation is being recorded.

DAY.CONVERSIONS.END, MO.CONVERSIONS.END, YR.CONVERSIONS.END - The date at which conversions (under ideal conditions) should be finished.

DESTROYED - Records the number of engines destroyed due to testing.

EARLY.OVERHAUL.OK - A capability to allow engines which have not completed their warranted time on base to be serviced early. This could result in smoother depot work loads. This capability is not used in the initial simulation.

END.DAY, END.MO, END.YR - The ending date of the simulation.

ENGINE.SEQUENCE.NUMBER - Maintains the ENGINE.ID.NUMBER of the last engine produced. Used to sequence the ENGINE.ID.NUMBERs as engines are produced.

EVIP.TEST, OTL.TEST - The length of time required for EVIP and OTL testing.

FAIL.RATE - Percentage of engines which will fail at some point in their warranted lifetime.

FULL.TIME - See CONVERSION.TIME.

LIMITED.TIME - See CONVERSION.TIME.

MONTH.CONVERSIONS.END - See DAY.CONVERSIONS.END

NUMBER.BASES - The number of bases where ALCMs are deployed in the simulation.

NUMBER.SPARES - The number of spare engines provided to support the system.

NUM.BROKEN, NUM.OVERDUE, NUM.SPARE - Stores the number of engines that are currently broken, overdue, or being used as or requiring a spare.

OCALC.CAPACITY, WILLIAMS.CAPACITY - The number of work stations established at the depots. This is the number of engines that can be serviced by the depot at the same time.

OPEN.DAY, OPEN.MO, OPEN.YR - The date Oklahoma City ALC (OCALC) begins depot operations.

OTL.TEST - See EVIP.TEST

RECOVERY.FAILURE.RATE - Percentage of OTL test launches which are not recovered.

REFURB.TIME - See CONVERSION.TIME

START.DATE - The date at which an engine begins its warranted lifetime.

START.DAY, START.MO, START.YR - The beginning date of the simulation.

STREAM1, STREAM2, STREAM3, STREAM4, STREAM5 - Values used to initialize the random number streams.

TOTAL.MONTHS - The number of months the simulation runs.

TRANSPORT.DAYS - The number of days required to transport an engine from one location to another.

WILLIAMS.CAPACITY - See OCALC.CAPACITY

YR.CONVERSIONS.END - See DAY.CONVERSIONS.END

Appendix G: Sample Verification Process

This appendix contains one example of the procedure used by the authors to verify the correct operation of the SIMSCRIPT model. Page 125 shows one of the PROCESSes (ENTER.BASE) from a modified program used for verification. This PROCESS, as did all the others, had verification statements inserted in various places. These verification statements were designed to monitor appropriate variables (those that affected the program's logic decisions) both before and after the processing of a specific engine. Examination of system variables and engine attributes at PROCESS initiation can tell the modeler what direction the engine should take at each decision point, if the model operates as planned. If the engine actually followed the correct path through the PROCESS, certain system variables and engine attributes would be changed. An examination of these variables and attributes at the end of the PROCESS will reveal if the correct changes have been made. Noting the correct changes verifies the operation of that portion of the PROCESS.

Prior to the verification runs, another modified program was run which "captured" and printed the ENGINE.ID.NUMBER(s) of those engines passing through each possible "logic path" of the model. These engines were subsequently traced through the model by the verification program to determine the model's correct operation through the logic paths taken by each engine. Enough engines were traced to verify the correct operation of all possible logic paths and path combinations.

This analysis did uncover several minor logic errors in the model. These errors were then corrected and more verification runs made on those logic paths, verifying their correct operation.

The example presented traces engine number 911 (memory location 186573) as it is being removed and replaced from an operational missile, in preparation for a limited overhaul, after exceeding its warranted lifetime on base. Line number 4 of ENTER.BASE calls the routine MISTEAK which examines the engine to determine if it is the one being traced (i.e., ENGINE.ID.NUMBER = 911). If it is, MISTEAK prints out a list of system variables and engine attributes (see lines 1-46 on page 124) and a code (see line 46 on page 124) identifying the calling statement's location in the program. This procedure is repeated (see lines 1-47 on page 126) in line 52 of ENTER.BASE, giving the modeler a "before" and "after" picture of the system.

As seen from the initial variable list on page 124, N.TAKE.SET (the number of engines in the TAKE.SET) is equal to zero. With proper operation, lines 7-9 of ENTER.BASE should be processed transferring control to line 29. This operation is verified by observing the assignment of 186573 (the memory location of FLAG) to ENG.OUT, allowing ENG.OUT to trigger routine MISTEAK in line 52 of ENTER.BASE. Line 8 has also correctly changed engine 911's CLAIM attribute from 70 (TAKEN) to 600 (NONE). If the alternate path in ENTER.BASE (lines 11-28) had been incorrectly taken, a verification statement in line 23 would have printed out, which did not happen.

The initial values of 911's attributes TEST.STATUS (600 = NONE) and STATUS (610 = OVERDUE), indicate that the engine should bypass statements 31 and 32 and activate statement 34 of ENTER.BASE. This action was verified by observing the activation of the process TRANSPORT with engine 911, immediately after the end of the ENTER.BASE process. Statement 37 was verified by observing the change in the number of engines listed as overdue (NOVERDUE) from 2 to 1, and observing no change (0 to 0) in the number of broken engines (NBROKEN). Line 47's operation was verified by observing the change in 911's ALERT.MISSILE.SET membership from 1 (in the set) to 0 (not in the set). Finally, line 49 was verified by observing the change in 911's STATUS from 610 (OVERDUE) to 600 (NONE).

VARIABLES BEFORE PROCESSING

```
 1 MONTH.........................................2
 2 DEPOT1 WAIT QUEUE.............................0
 3 DEPOT2 WAIT QUEUE.............................0
 4 ENGINES BEING SERVICED BY DEPOT1..............18
 5 ENGINES BEING SERVICED BY DEPOT2..............0
 6 REPAIRED ENGINES AVAILABLE, DEPOT1............88
 7 REPAIRED ENGINES AVAILABLE, DEPOT2............0
 8 NUMBER OF ENGINES BEING TESTED................2
 9 NUMBER OF ENGINES ON ALERT....................1525
10 NUMBER OF SPARE ENGINES REQUIRED..............28
11 NUMBER OF ENGINES IN TAKE.SET.................0
12 NUMBER OF ENGINES BROKEN......................0
13 NUMBER OF ENGINES OVERDUE.....................2
14 NUMBER OF LIMITED CONVERSIONS AVAILABLE.......0
15 NUMBER OF FULL CONVERSIONS AVAILABLE..........0
16
17
18 PROCESS NUMBER..............4
19 SIMULATION TIME.............1880
20 LIFETIMES DATED.............2818
21 NUMBER  FAILED..............406
22 NUMBER  DESTROYED...........18
23 JTA'S   DESTROYED...........16
24
25
26 ATTRIBUTES OF ENGINE CALLED ENG:
27 ARRIVAL.DATE..................1293
28 CLAIM.........................70
29 ENGINE.ID.NUMBER..............911
30 EXPIRATION.DATE...............1880
31 LOCATION......................3
32 RANK.DATE.....................1880
33 SHIPPING.STATUS...............300
34 START.DATE....................967
35 SPARE.STATUS..................650
36 STATUS........................610
37 TEST.STATUS...................600
38 TYPE..........................101
39 TYPE.SERVICE..................100
40 IN PRODUCTION POOL?...........0
41 IN TEST.SET?..................0
42 IN REPAIRED.SET?..............0
43 IN TAKE.SET?..................0
44 IN ALERT.MISSILE.SET?.........1
45
46 DEBUG LOCATION..............33
```

```
PROCESS ENTER.BASE(ENG.IN,FLAG)
DEFINE ENG.IN,ENG.OUT,FLAG AS INTEGER VARIABLES
CALL MISTEAK(ENG.IN,32)
CALL MISTEAK(FLAG,33)
LET SHIPPING.STATUS(ENG.IN) = NONE
IF  N.TAKE.SET = 0
  LET ENG.OUT = FLAG
  LET CLAIM(ENG.OUT) = NONE
  GO START
ALWAYS
LET CLAIM(FLAG) = NONE
IF FLAG IS NOT IN TAKE.SET AND STATUS(FLAG) NE NONE
  FILE THIS FLAG IN TAKE.SET
ALWAYS
FOR EACH ENGINE OF TAKE.SET, WITH LOCATION(ENGINE) =
LOCATION(FLAG) AND CLAIM(ENGINE) NE TAKEN, FIND ENG.OUT =
ENGINE, IF FOUND
  IF ENGINE.ID.NUMBER(ENG.OUT) = INNUM
    FOR EACH ENGINE OF TAKE.SET, UNTIL ENGINE = ENG.OUT
    DO
      PRINT 1 LINE WITH ENGINE.ID.NUMBER(ENGINE),
      LOCATION(ENGINE), CLAIM(ENGINE) THUS
      #4 EID *******   LOC ****   CLAIM ****
    LOOP
  ALWAYS
  GO START
ALWAYS
LET ENG.OUT = FLAG
'START'
IF TEST.STATUS(ENG.OUT)=EVIP AND STATUS(ENG.OUT) NE BROKE
  CALL SHIP GIVING ENG.OUT
  SCHEDULE A TEST(ENG.OUT) IN TRANS.DAYS DAYS
ELSE
  SCHEDULE A TRANSPORT(ENG.OUT) NOW
ALWAYS
IF STATUS(ENG.OUT) = DUE
  SUBTRACT 1 FROM NOVERDUE
ELSE
  IF STATUS(ENG.OUT) = BROKE
    SUBTRACT 1 FROM NBROKEN
  ALWAYS
ALWAYS
IF ENG.OUT IS IN TAKE.SET
  REMOVE THIS ENG.OUT FROM TAKE.SET
ALWAYS
IF ENG.OUT IS IN ALERT.MISSILE.SET
  REMOVE THIS ENG.OUT FROM ALERT.MISSILE.SET
ALWAYS
LET STATUS(ENG.OUT) = NONE
LET LOCATION(ENG.IN) = LOCATION(ENG.OUT)
FILE THIS ENG.IN IN THE ALERT.MISSILE.SET
CALL MISTEAK(ENG.IN,34)    CALL MISTEAK(ENG.OUT,35)
END
```

VARIABLES AFTER PROCESSING

```
 1 MONTH.........................................2
 2 DEPOT1 WAIT QUEUE.............................0
 3 DEPOT2 WAIT QUEUE.............................0
 4 ENGINES BEING SERVICED BY DEPOT1..............18
 5 ENGINES BEING SERVICED BY DEPOT2..............0
 6 REPAIRED ENGINES AVAILABLE, DEPOT1............88
 7 REPAIRED ENGINES AVAILABLE, DEPOT2............0
 8 NUMBER OF ENGINES BEING TESTED................2
 9 NUMBER OF ENGINES ON ALERT....................1525
10 NUMBER OF SPARE ENGINES REQUIRED..............28
11 NUMBER OF ENGINES IN TAKE.SET.................0
12 NUMBER OF ENGINES BROKEN......................0
13 NUMBER OF ENGINES OVERDUE.....................1
14 NUMBER OF LIMITED CONVERSIONS AVAILABLE.......0
15 NUMBER OF FULL CONVERSIONS AVAILABLE..........0
16
17
18 PROCESS #...................
19 SIMULATION TIME.............1880
20 LIFETIMES DATED.............2818
21 NUMBER  FAILED..............406
22 NUMBER  DESTROYED...........18
23 JTA'S   DESTROYED...........16
24
25
26
27 ATTRIBUTES OF ENGINE CALLED ENG:
28 ARRIVAL.DATE..................1293
29 CLAIM.........................600
30 ENGINE.ID.NUMBER..............911
31 EXPIRATION.DATE...............1880
32 LOCATION......................3
33 RANK.DATE.....................1880
34 SHIPPING.STATUS...............600
35 START.DATE....................947
36 SPARE.STATUS..................650
37 STATUS........................600
38 TEST.STATUS...................600
39 TYPE..........................101
40 TYPE.SERVICE..................100
41 IN PRODUCTION POOL?...........0
42 IN TEST.SET?..................0
43 IN REPAIRED.SET?..............0
44 IN TAKE.SET?..................0
45 IN ALERT.MISSILE.SET?.........0
46
47 DEBUG LOCATION..............35
```

Bibliography

1. Air Force Logistics Command. Initial Requirements Determination. AFLCR 57-27. Wright-Patterson AFB: HQ AFLC, 20 July 1982.

2. Berman, Morton B. Notes on Validating/Verifying Computer Simulation Models. Rand Paper P-4891. Rand Corporation, Santa Monica CA, August 1972 (AD-755 078).

3. Box, G. E. P. "Some Theorems on Quadratic Forms Applied in the Study of Analysis of Variance Problems, II. Effect of Inequality of Variance and of Correlation of Errors in the Two-Way Classification," The Annals of Mathematical Statistics, 25 (3): 484-498 (September 1954).

4. Cox, Larry W. and Dr. Jacques S. Gansler. "Evaluating the Impact of Quantity, Rate, and Competition," Concepts, 4 (4): 29-53 (Autumn 1981).

5. Department of the Air Force. Acquisition Program Management. AFR 800-2. Washington: HQ USAF, 13 August 1982.

6. Department of Defense. Determination of Initial Requirements for Secondary Item Spare and Repair Parts. DOD Instruction 4140.42. Washington: Government Printing Office, 7 August 1974.

7. Drezner, Stephen M. and Richard J. Hillestad. Logistics Models: Evolution and Future Trends. Rand Paper P-6748. Rand Corporation, Santa Monica CA, March 1982 (AD-118 808).

8. Emory, William C. Business Research Methods. Homewood IL: Richard D. Erwin, Inc., 1980.

9. Garratt, Michael. "Statistical Validation of Computer Models," Proceedings of the 1974 Summer Computer Simulation Conference. 915-927. The Simulation Councils, Inc., La Jolla CA, 1974.

10. Gilmour, Peter. "A General Validation Procedure for Computer Simulation Models," The Australian Computer Journal, 5 (3): 127-131 (1973).

11. Kleijnen, Jack P. C. Statistical Techniques in Simulation, Part I. New York: Marcel Dekker, Inc., 1975.

12. Kleijnen, Jack P. C. Statistical Techniques in Simulation, Part II. New York: Marcel Dekker, Inc., 1975.

13. Nolan, Richard L. "Verification /Validation of Computer Simulation Models," Proceedings of the Summer Computer Simulation Conference. 1254-1265. The Simulation Councils, Inc., La Jolla CA, 1972.

14. Price, Bernard C. "Alternatives in Provisioning Computational Methodology," Logistics Spectrum, 15 (4):14-19 (Winter 1981).

15. Russell, Edward C. Building Simulation Models with SIMSCRIPT II.5. Los Angeles: C.A.C.I., Inc.-Federal, 1983.

16. Schoderbek, Charles G., Peter P. Schoderbek and Asterios G. Kefalas. Management Systems Conceptual Considerations. Dallas: Business Publications, Inc., 1980.

17. Shannon, Robert E. Systems Simulation the Art and Science. Englewood Cliffs NJ: Prentice-Hall, Inc., 1975.

18. -----. SIMSCRIPT II.5 Programming Language. Los Angeles: C.A.C.I., Inc.-Federal, 1983.

19. -----. SIMSCRIPT II.5 Reference Handbook (Second Edition), edited by Jay E. Braun. Los Angeles: C.A.C.I., Inc.-Federal, 1983.

20. Uhrig, Robert., Engine Logistics Office. Personal interviews. Aeronautical Systems Division, Wright-Patterson AFB OH, 3 January through 29 August 1984.

21. Voosen, B. J. Simulation and Evaluation of Logistics Systems, Rand Memorandum RM-4589-PR. Rand Corporation, Santa Monica CA, 1965 (AD-471 460).

VITAE

Captain Doug Rickard was born on 27 June 1947 in San Francisco, California. After completing high school in Salinas, California, he attended Hartnell Jr. College, then enlisted in the USAF in December of 1967. He was trained as a Missile Guidance and Control specialist with tours at Nellis AFB, NV; Homestead AFB, FL; and Hahn AB, Germany. He was accepted for the Airman Education and Commissioning program while in Germany and obtained a Bachelor of Science degree in Management Science from Oklahoma State University in 1973. He then trained as a navigator and received his wings at Mather AFB, CA in 1974, and was assigned to Norton AFB, CA as a C-141 navigator. In 1976 he was selected for pilot training and received his wings at Reese AFB, TX. He was assigned to the Mather AFB, CA again, this time to fly the B-52G. He served as a squadron copilot, instructor copilot, STAN EVAL copilot, aircraft commander and as the Chief of Bomber Scheduling while at Mather AFB, with a break in 1979 to attend Squadron Officer School at Maxwell AFB, AL. In May of 1983 he was assigned to the School of Systems and Logistics, Air Force Institute of Technology.

Captain Rickard is married to the former Gail Sue Emerson of San Bernardino, California. They have five children: Robert, Alisa, Michael, Julie and Christopher.

Permanent address: 1315 12th St.
Los Osos, Calif. 93401

Captain Thomas G. Schommer was born 17 December 1953 in Omaha, Nebraska. He graduated from high school in Columbus Ohio in 1971 and attended Marian College in Indianapolis, Indiana receiving a degree of Bachelor of Arts i music in 1975. He received his commission in the USAF through OTS in 1979. He served as Material Management Officer for the 305th Supply Squadron as Grissom AFB, Indiana until September, 1981. He was reassigned to the HQ First Combat Evaluation Group, Barksdale AFB, Louisiana serving as the Material Officer for the Radar Bomb Scoring activity. He entered the School of Systems and Logistics, Air Force Institute of Technology, in May 1983.

Captain Schommer is married to the former Elvira Gonzalez of Philadelphia, Pennsylvania. They have three children: Jared, Hena Marie, and Manuel.

Permanent address: 5032 Victoria Rd.
Indianapolis, Indiana 46208

REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION
UNCLASSIFIED

1b. RESTRICTIVE MARKINGS

2a. SECURITY CLASSIFICATION AUTHORITY

2b. DECLASSIFICATION/DOWNGRADING SCHEDULE

3. DISTRIBUTION/AVAILABILITY OF REPORT
Approved for public release; distribution unlimited.

4. PERFORMING ORGANIZATION REPORT NUMBER(S)
AFIT/GLM/LSM/84S-55

5. MONITORING ORGANIZATION REPORT NUMBER(S)

6a. NAME OF PERFORMING ORGANIZATION
School of Systems and Logistics

6b. OFFICE SYMBOL *(If applicable)*
AFIT/LS

7a. NAME OF MONITORING ORGANIZATION

6c. ADDRESS *(City, State and ZIP Code)*
Air Force Institute of Technology
Wright-Patterson AFB, Ohio 45433

7b. ADDRESS *(City, State and ZIP Code)*

8a. NAME OF FUNDING/SPONSORING ORGANIZATION

8b. OFFICE SYMBOL *(If applicable)*

9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER

8c. ADDRESS *(City, State and ZIP Code)*

10. SOURCE OF FUNDING NOS.

PROGRAM ELEMENT NO.	PROJECT NO.	TASK NO.	WORK UNIT NO.

11. TITLE *(Include Security Classification)*
See Box 19

12. PERSONAL AUTHOR(S)
Douglas P. Rickard, B.S., Capt, USAF and Thomas G.Schommer, B.A., Capt, USAF

13a. TYPE OF REPORT
MS Thesis

13b. TIME COVERED
FROM ________ TO ________

14. DATE OF REPORT *(Yr., Mo., Day)*
1984 September

15. PAGE COUNT
141

16. SUPPLEMENTARY NOTATION

Approved for public release: IAW AFR 190-17.
[illegible] 14 Sept 84
LYNN E. WOLAVER
Dean for Research and Professional Development
Air Force Institute of Technology (AFC)
Wright-Patterson AFB OH 45433

17. COSATI CODES

FIELD	GROUP	SUB. GR.
15	05	

18. SUBJECT TERMS *(Continue on reverse if necessary and identify by block number)*
Computerized Simulation, Cruise Missiles, Engines, Logistics, Spare Parts, Systems Management

19. ABSTRACT *(Continue on reverse if necessary and identify by block number)*

Title: A SIMULATION MODEL FOR AIR LAUNCHED CRUISE MISSILE ENGINE MANAGEMENT

Thesis Chairman: Dr. Panna B. Nagarsenker

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT
UNCLASSIFIED/UNLIMITED [✓] SAME AS RPT. ☐ DTIC USERS ☐

21. ABSTRACT SECURITY CLASSIFICATION
UNCLASSIFIED

22a. NAME OF RESPONSIBLE INDIVIDUAL
Panna B. Nagarsenker, Associate Prof.

22b. TELEPHONE NUMBER *(Include Area Code)*
513-255-7210

22c. OFFICE SYMBOL
AFIT/ENC

This paper examines the management of spares acquisition and logistics support activities for the Air Launched Cruise Missile (ALCM) engine. A SIMSCRIPT II.5 simulation model of the ALCM system is developed and the probable ranges of five relevant factors (transportation time, maintenance duration, engine failure rate, test duration and test loss rate) are determined. The model is manipulated using a factorial design with the five factors at the extremes of their ranges. An ANOVA is used to determine if changes in these five factors significantly impact the operational capability of the ALCM engine. This capability is expressed as the average number of days an engine must be used as an operational asset past the manufacturer's warranty period without an overhaul. The ANOVA indicated that changes in transportation time and maintenance duration had a significant effect on the operational capability of the ALCM engine and thus required further investigation to refine their range of values.

The model's operation, verification, input requirements and output capabilities were documented so that the model could be used as a management tool when validation is completed. This documentation will give other modelers the depth of understanding necessary to adapt the model to their particular use. The model was designed with the flexibility necessary to modify the assumptions and limitations built into it so the model could "grow" as the ALCM system evolves.

The model is used

ALCM engine managers should eventually be able to use the model to test the effects (on a number of variables of interest) of changes in maintenance policies and timing. The model should easily be adaptable to reflect the environment of the Ground Launched Cruise Missile and the Submarine Launched Cruise Missile systems. It could then be used to estimate the number of spare engines required in these systems to meet various engine capability levels and other management goals.